

Dr. Dobb's

SOFTWARE
TOOLS FOR
PROFESSIONAL
PROGRAMMER

**DDJ's
ENCRYPTION
CONTEST!**

See Page 40

JOURNAL

UNLOCKING ENCRYPTION ALGORITHMS

- BLOCK ENCRYPTION
- LUCAS FUNCTIONS
- SECURE HASH ALGORITHM

WAVELET PACKET
TRANSFORMS

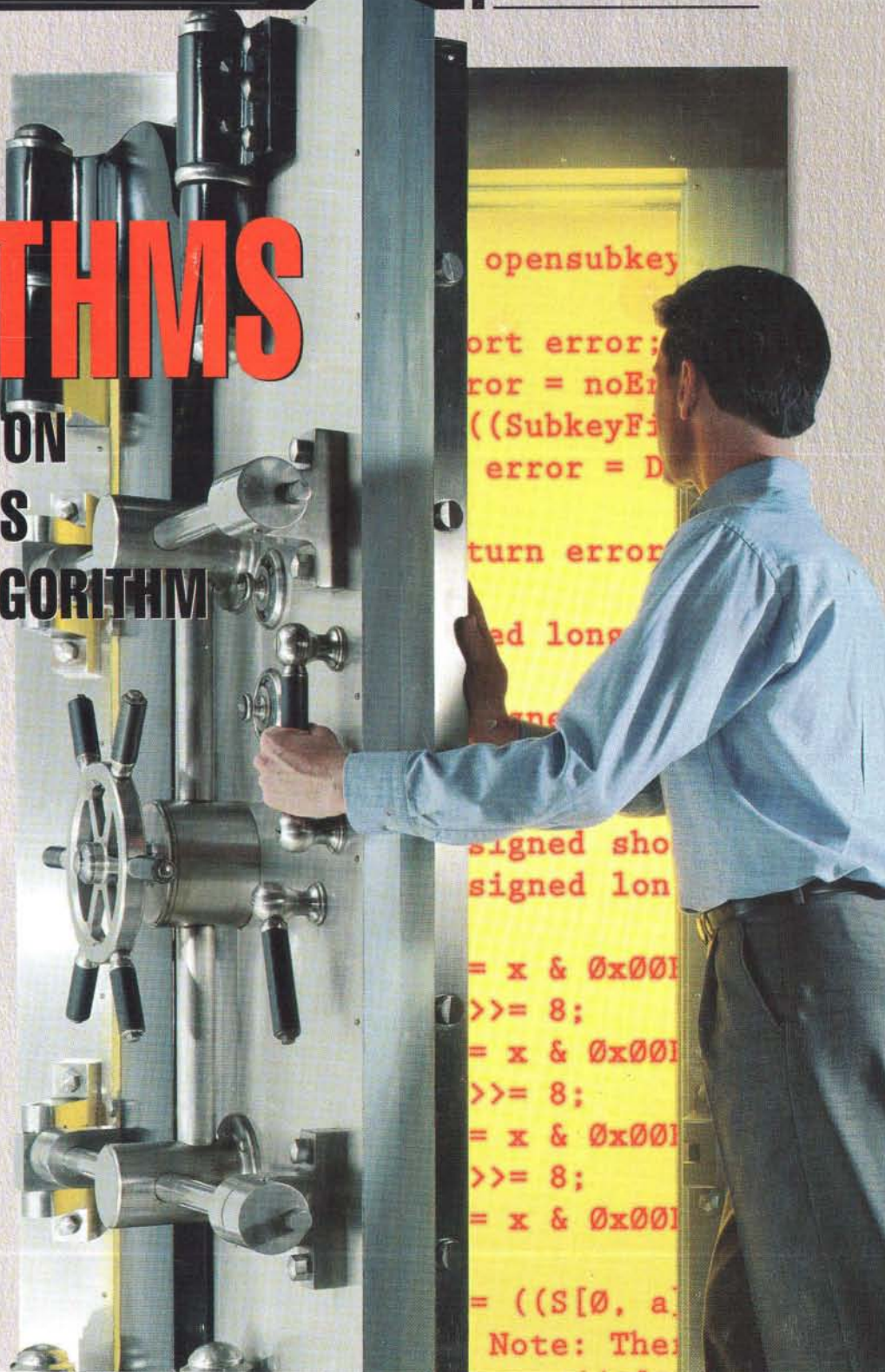
DATA ACQUISITION

EXAMINING
WINDOWS HELP
AUTHORING TOOLS

ECHONETS

and more!

\$3.95 (\$4.95 CANADA)



Periodic Table of Visual Components

Diagram illustrating the structure of a component box:

- 5**: Component Number
- C++**: Symbol
- Visual C++ 1.5**: Proper Name

			IIIB			IVB			VB		
			5			6			7		
			C++			BOOKS			CCMD		
			Visual C++ 1.5			Books Online			CCmdTarget		
			13			14			15		
			APP			MFC			CWIN		
			App Studio			Microsoft Foundation Class Library 2.5			CWinApp		
IIA			IB			IIB					
27			28			29			30		
CB			OLE			ODBC			DDX		
ControlBar			Object Linking & Embedding 2.0			Open Database Connectivity			Dialog Data Exchange		
									31		
									APW		
									AppWizard		
									32		
									CLW		
									ClassWizard Controls		
									33		
									CDOC		
									CDocument		
									45		
									TB		
									46		
									COLE		
									47		
									CREC		
									CRecordSet		
									48		
									DDV		
									Dialog Data Validation		
									49		
									HELP		
									50		
									CFRM		
									CFormView		
									51		
									CWND		
									CWnd		
									82		
									83		

THE BUILDING BLOCKS OF VISUAL DEVELOPMENT.

The elements of next generation applications are found in the components of the Microsoft® Visual C++™ development system, version 1.5 today.

Innovative *wizard* technology allows you to seamlessly meld components into sophisticated applications, in weeks not months. Just remember this simple equation: MFC 2.5, OLE 2.0, ODBC.

That's a perfect formula for building and reusing components. At the nucleus is MFC 2.5 (Microsoft Foundation Class Libraries), the standard API for Windows.™ The new OLE (Object Linking and Embedding) classes make it easier than ever to implement OLE's unprecedented integration power, new drag-and-drop features, visual editing and OLE automation. In all, it provides more than 19,000 lines of code you want, but won't have to write or rewrite.

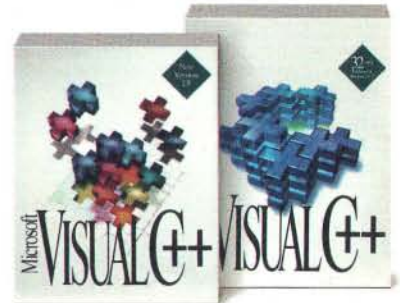
In addition, the new ODBC (Open Database Connectivity)

classes provide access to major databases and data binding without coding. It's a powerful new platform for creating high-performance database applications.

Best of all, your 16-bit MFC applications are portable to the Windows NT™ operating system and future versions of Windows. Now that's good chemistry.

If you want the latest tools for the latest Windows technology, ask for Microsoft Visual C++ 1.5.

Call us now at (800) 434-3980 and we'll make you a catalyst for change today and into the future of Windows operating systems.



New 16-bit version 1.5 joins the Visual C++ family of development systems for Windows and Windows NT.

Microsoft

FEATURES

THE CAMBRIDGE ALGORITHMS WORKSHOP

by Bruce Schneier

Some of the best and the brightest in the world of cryptography gathered at Cambridge University to challenge each other with new algorithms designed to run quickly in software. Bruce, who presented a paper at the workshop, reports on the conference, as well as on the current state of encryption technology in general.

CRYPTOGRAPHY WITHOUT EXPONENTIATION

by Peter Smith

Peter, who presented LUC public-key encryption in *DDJ* over a year ago, extends the algorithm by adding three new cryptosystems: a Lucas-function El Gamal public-key encryption, a Lucas-function El Gamal digital signature, and a Lucas-function-based key-negotiation method called LUCDIF.

SHA: THE SECURE HASH ALGORITHM

by William Stallings

The Secure Hash Algorithm (SHA), based on Ron Rivest's MD4 algorithm and developed by the National Institute of Standards and Technology, can be used in any security application that requires a hash code.

THE BLOWFISH ENCRYPTION ALGORITHM

by Bruce Schneier

Blowfish, a new block-encryption algorithm for 32-bit microprocessors, is designed to be fast, compact, simple, secure, and robust. Break it, and you can be the winner of our cryptography contest!

THE WAVELET PACKET TRANSFORM

by Mac A. Cody

The discrete wavelet transform is a subset of the far more versatile wavelet packet transform, which generalizes the time-frequency analysis of the wavelet transform. Mac presents a C implementation of the discrete wavelet transform algorithm.

FUZZY LOGIC IN C: AN UPDATE

by John A.R. Tucker, Phillip E. Fraley, Lawrence P. Swanson

In this article, our authors build upon Greg Viot's "Fuzzy Logic in C" by adding initialization, parsing, and output functions to provide a complete C implementation of fuzzy logic.

18

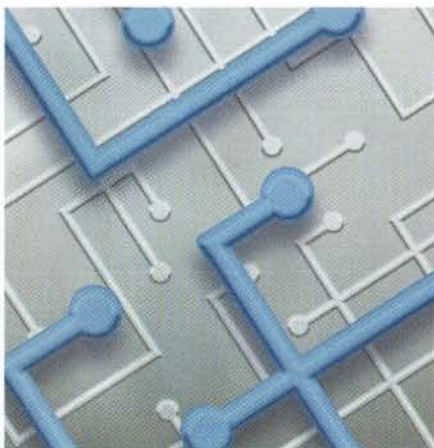
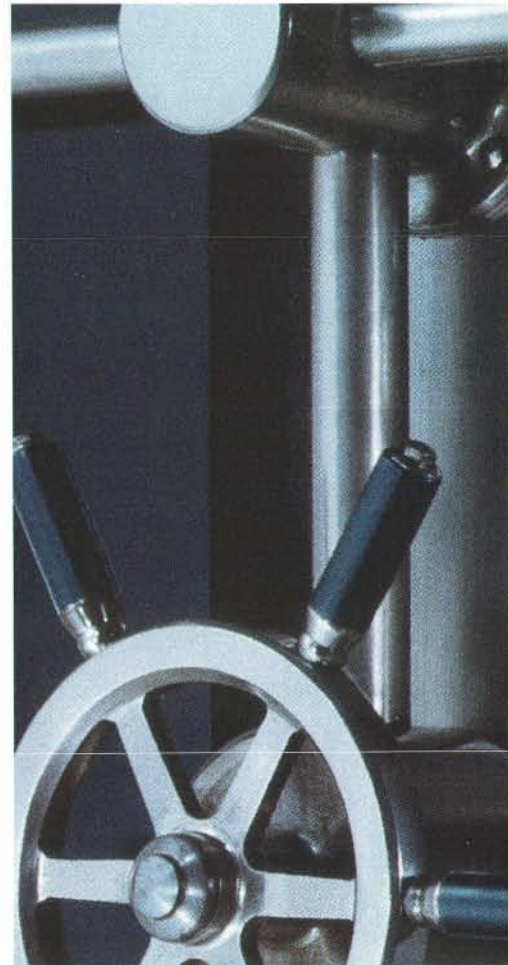
26

32

38

44

56



EMBEDDED SYSTEMS

DIGITAL I/O WITH THE PC

by Brian Hook and Dennis Shuman

You don't always have to resort to dedicated or expensive instruments for digital data acquisition. Brian and Dennis describe an integrated hardware/software system that enables digital I/O using a PC's parallel port.

64

NETWORKED SYSTEMS

ECHONETS, E-MEMES, AND EXTENDED REALITIES

72

by Scott B. Guthery

Mobile computing requires a new way of thinking about networks. Scott discusses the concept of switchless networks, called "echonets," and presents algorithms that make them possible.

EXAMINING ROOM

HELP FOR WINDOWS HELP AUTHORS

86

by Al Stevens

Eventually, every Windows developer has to build a help database. Al discusses what makes a good Windows help system and examines approaches and tools for creating them.

PROGRAMMER'S WORKBENCH

ALGORITHMS FOR DIRECTED GRAPHS

92

by Salvatore R. Mangano

Directed graphs underlie any tool that displays tree, class-relationship, or entity-relationship diagrams. Sal uses EOS, his C++ genetic-algorithm toolkit, and Visual C++ to create a Windows-hosted system for laying out directed graphs.

COLUMNS

PROGRAMMING PARADIGMS

109

by Michael Swaine

Michael examines how the British microcomputer revolution in the early 1980s led to the object-oriented model Apple's Newton uses today.

C PROGRAMMING

115

by Al Stevens

Borland's recent attempt to rewrite its software-license agreements didn't make anyone happy, especially programmers who use Borland tools.

ALGORITHM ALLEY

121

by Tom Swan

Tom presents an information-retrieval system based on the trie-search algorithm.

UNDOCUMENTED CORNER

125

edited by Andrew Schulman

In this month's "Undocumented Corner," Klaus Müller shows how to access the Windows internal instance-data structures, using a virtual device driver (VxD) loaded early in the Windows boot process, right after VMM.

PROGRAMMER'S BOOKSHELF

133

by Tom Ochs

Tom looks at two books on algorithm design and implementation—*Programming Classics: Implementing the World's Best Algorithms* and *Algorithms from P to NP*.



FORUM

EDITORIAL

6

by Jonathan Erickson

LETTERS

10

by you

SWAINE'S FLAMES

152

by Michael Swaine

PROGRAMMER'S SERVICES

OF INTEREST

148

by Monica E. Berg

SOURCE CODE AVAILABILITY

As a service to our readers, all source code is available on a single disk and online. To order the disk, send \$14.95 (California residents add sales tax) to *Dr. Dobb's Journal*, 411 Borel Ave., San Mateo, CA 94402, call 1-415-358-9500, x240, or use your credit card to order by fax, 1-415-358-9749. Specify issue number and disk format. Code is also available through the DDJ Forum on CompuServe (type GO DDJ), via anonymous FTP from site ftp.mv.com (192.80.84.1) in the /pub/ddj directory, and through DDJ Online, a free service accessible via direct dial at 1-415-358-8857 (1200/2400/9600 baud, 8-N-1).

NEXT MONTH

In May, we look under the hood of operating systems and microkernels including DOS, OS/2, Windows, UNIX, NetWare, and more.

Save Disk Space



PKZIP version 2.0

PC WORLD



WORLD CLASS AWARD

PKWARE® introduces the next generation of its award winning compression utility. PKZIP 2.0 yields greater performance levels than achieved with previous releases of the software. PKZIP compresses and archives files. This saves disk space and reduces file transfer time.

Software developers! You can significantly reduce product duplication costs by decreasing the number of disks required to distribute your applications. Call for Distribution License information.

Put Your Executables on a Diet

Software developers! Save disk space and media costs with smaller executables. You can distribute your software in a compressed form with PKLITE Professional™. PKLITE Professional gives you the ability to compress files so that they cannot be expanded by PKLITE™. This discourages reverse engineering of your programs.



PKLITE increases your valuable disk space by compressing DOS executable (.EXE and .COM) files by an average of 45%. The operation of PKLITE is transparent, all you will notice is more available disk space!

Compression for YOUR Application



The PKWARE Data Compression Library™ allows you to incorporate data compression technology into your software applications. The application program controls all the input and output of data, allowing data to be compressed or extracted to or from any device or area of memory.

All Purpose Data Compression Algorithm compresses ASCII or binary data quickly. The routines can be used with many popular DOS languages. A Windows DLL and an OS/2 32-bit version is also available!

PKWARE[®] INC.

The Data Compression Experts[®]

9025 N. Deerwood Drive Brown Deer, WI 53223-2437
(414) 354-8699 Fax (414) 354-8559

PKWARE Data Compression Library for DOS \$275 PKWARE Data Compression Library for OS/2 \$350
PKWARE Data Compression Library DLL for Windows \$350
PKZIP \$47.00 PKLITE \$46.00 PKLITE Professional \$146.00
Please add \$5.00 S&H per package in the US & Canada, \$11.25 overseas.
Wisconsin residents add appropriate state sales tax & county sales tax.
Visa and Mastercard accepted, no COD orders.

DD4-94

CIRCLE NO. 539 ON READER SERVICE CARD

Dr. Dobb's[®] SOFTWARE TOOLS FOR THE PROFESSIONAL PROGRAMMER JOURNAL

VICE PRESIDENT-PUBLISHER Peter Hutchinson

EDITORIAL

EDITOR-IN-CHIEF Jonathan Erickson
MANAGING EDITOR Tami Zemel
EXECUTIVE EDITOR Michael Floyd
SENIOR TECHNICAL EDITOR Ray Valdés
SENIOR PRODUCTION EDITOR Monica E. Berg
ASSOCIATE EDITOR John Dorsey
ASST. MANAGING EDITOR Christine de Chutkowski
ART DIRECTOR Michael Hollister
CONTRIBUTING EDITORS Al Stevens, Tom Genereaux,
Andrew Schulman, Ray Duncan, David Betz, Tom Swan
EDITOR-AT-LARGE Michael Swaine
PRODUCTION MANAGER Amy Schulman Lesovsky
COVER PHOTOGRAPHER Michael Carr

CIRCULATION

DIRECTOR OF CIRCULATION Jerry Okabe
CIRCULATION MANAGER Michael Poplaro

ADMINISTRATION

ACCOUNTING SUPERVISOR Renate Kernke
ACCOUNTS RECEIVABLE Wendy Ho

MARKETING/ADVERTISING

ASSOCIATE PUBLISHER Cynthia C. Sandor
ADVERTISING COORDINATOR Sara D. Wood
DDJ BUSINESS MANAGER Lisa Tarlton
MARKETING MANAGERS Valerie Dow, Mike Kuehl
ACCOUNT MANAGERS see page 136

MILLER FREEMAN INC.

CHAIRMAN OF THE BOARD Graham J.S. Wilson
PRESIDENT/CEO Marshall W. Freeman
EXECUTIVE VP/COO Thomas L. Kemp
SENIOR VICE PRESIDENTS H. Verne Packer, Wini D. Ragus, Donald A. Pazour
VICE PRESIDENT/SOFTWARE Regina Starr Ridley
VICE PRESIDENT/CFO Warren Ambrose
VICE PRESIDENT/ADMIN. Charles H. Benz
VICE PRESIDENT/PRODUCTION Andrew A. Mickus
VICE PRESIDENT/CIRCULATION Jerry Okabe

mf Miller Freeman, Inc.
A publication of M&T Publishing, division of Miller Freeman Inc., member of the United Newspapers Group.

DR. DOBB'S JOURNAL (USPS 307690) is published monthly by Miller Freeman, Inc., 411 Borel Ave., San Mateo, CA 94402-3522; 415-358-9900. Second-class postage paid at San Mateo and at additional entry points.

ARTICLE SUBMISSIONS: Send manuscripts and disk (with article, listings, and letter to the editor) to *DDJ* Submissions, 411 Borel Ave., San Mateo, CA 94402-3522.

DDJ ON COMPUSERVE: Type GO DDJ.

SUBSCRIPTION: \$29.97 for 1 year; \$56.97 for 2 years. Foreign orders must be prepaid in U.S. funds drawn on a U.S. bank. Canada and Mexico: \$45.00 per year. All other foreign: \$70.00 per year.

FOR SUBSCRIPTION QUESTIONS, change of address, and orders, call toll-free 800-456-1215 (U.S. and Canada). For all other countries, call 303-447-9330, or fax 303-443-5080. On CompuServe, the customer-service number is 71572.341. Or write, *Dr. Dobb's Journal*, P.O. Box 56188, Boulder, CO 80322-6188.

POSTMASTER: Send address changes to *Dr. Dobb's Journal*, P.O. Box 56188, Boulder, CO 80322-6188. **ISSN 1044-789X.** **GST (Canada) # R124771239.**

FOREIGN NEWSSTAND DISTRIBUTOR: Worldwide Media Service Inc., 30 Montgomery St., Jersey City, NJ 07302; 212-332-7100.

Entire contents copyright © 1994 by Miller Freeman, Inc., unless otherwise noted on specific articles. All rights reserved.

ABP
American Business Press

Printed in the
USA

The Audit Bureau

New 6.1
Now Available

VISUALIZE THE PERFECT WINDOWS DEVELOPMENT ENVIRONMENT.

Just imagine a world without slow compilers or slow, fat code. Symantec C++ Professional 6.1 gives you everything you could want in a perfect environment.

The perfect compiler.

We've made Symantec C++ Professional faster and more efficient.

And, our one-pass compiler isn't just faster — it produces faster applications. Thanks to powerful global and local optimizations. And some unique C++ optimizations you won't find anywhere else.

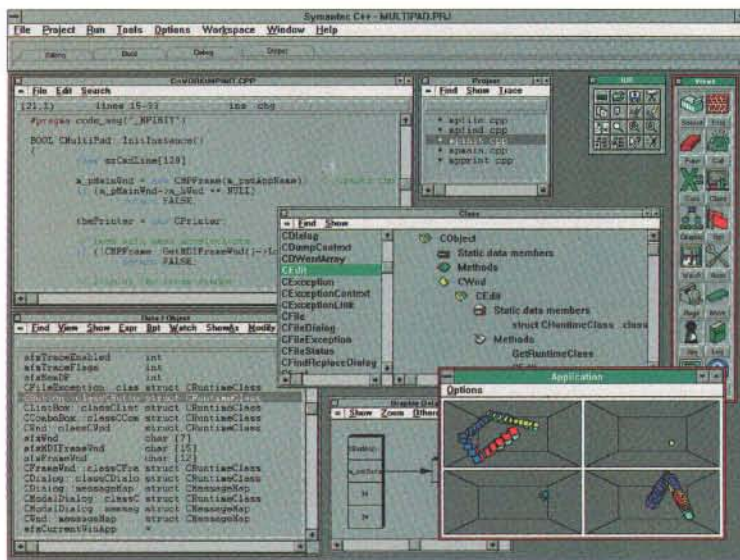
The perfect environment.

Symantec C++ is the first development system to fully integrate editing, compiling, debugging and version control in a completely new and graphical Integrated Development and Debugging Environment (IDDE). This breakthrough new IDDE is packed with graphical tools, a great new resource editor and extensive drag-and-drop capabilities.

The perfect visual tools.

And the best linker.

Using technology developed exclusively for Symantec by Blue Sky Software, Symantec C++



gives you the world's most sophisticated visual development tools. Without writing a line of code, you can visually create toolbars, 3-D graphic buttons, dialog boxes, MDI windows and more. Just double click on any object to add code immediately.

In addition, we've added OPTLINK, absolutely the fastest linker you can buy.

Win32s from the leader in 32-bit compilers.

Win32s is free in every box. Create true 32-bit applications that run unchanged under

both Windows 3.1 and Windows NT. So you can develop and debug apps on your existing Windows system.

It's easy to switch.

We even provide a complete set of utilities that can automatically translate Microsoft and Borland command line options to Symantec C++. We've also included the full Microsoft Foundation Classes 2.0 and support for Microsoft Visual Basic controls.

Order today.

Symantec C++ Professional 6.1 is now priced at just \$499.* But if you are already a Microsoft or Borland user, we have a special upgrade offer of just \$199.95.** A 60% savings.

See your nearest dealer. Or call 1-800-453-1077 ext. 9A11 and order now. You'll find Symantec C++ Professional 6.1 is everything a Microsoft user or a Borland C++ user could ever want.



Symantec C++ is available in both Professional and Standard versions.

SYMANTEC.™

*Suggested retail price. Actual price may vary. **Upgrade offer plus shipping, handling and applicable tax. Offer price in U.S. dollars. Valid in U.S. and Canada only. For more information in Europe, call 31-71-353111. In Australia, call 61-2-879-6577. In Canada, call 1-800-667-5661. Everywhere else, call 408-252-3570. Symantec C++ is a registered trademark of Symantec Corporation. All other products are trademarks or registered trademarks of their respective holders.

©1994 Symantec Corporation. All rights reserved.

CIRCLE NO. 131 ON READER SERVICE CARD

Ten Years After

Until the Tonya Harding/Nancy Kerrigan brouhaha ushered in the notion of tag-team figure skating, it had been at least ten years since anything really interesting happened in ice skating. Even back in the 1984 Winter Olympics, what held the attention of anyone other than hardcore figure-skating devotees was whether or not U.S. figure skater Scott Hamilton's jeans could endure one more Russian split, or if a triple axel would finally dislodge East German Katarina Witt's Jimmy Johnson-like coiffure. In retrospect, it would have been far more interesting had we been able to foresee the current plight of the '84 Winter Olympics host city—Sarajevo, Yugoslavia. And speaking of Russian splits, not only has the past decade brought us the breakup of the U.S.S.R., but the demise of East Germany, Yugoslavia, and other such countries as well.

In the relatively calm world of computer programming, it's hard to believe that ten years have passed since Bjarne Stroustrup introduced the C++ programming language. In the high-octane software-development arena, we've come to believe that things move fast, even though it's taken a decade for C++ to gain little more than a toehold. Stroustrup ostensibly created C++ "to make writing good programs easier and more pleasant for the individual programmer." Still, it's taken the marketing might of companies such as AT&T, Microsoft, Sun, IBM, and Borland to make the language a commercial success. The fact that Smalltalk has been around more than twice as long as C++ with far less success, or that there's so darn much Cobol code still out there, says something about the software-development community's desire to absorb and adopt new languages, not to mention the marketing efforts of C++ vendors.

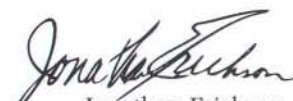
As it enters its second decade, C++ finally seems entrenched. But even though the language is being used to program everything from PCs to supercomputers, the jury is still out on whether or not Stroustrup's goal of making programmers more productive and software less complex has been—or can be—achieved. The real lesson to be learned here is that software development is a process which does not adapt to change easily. There's almost always too much at stake for programmers to casually pick up one language while discarding another—other than for educational or entertainment reasons. Instead, software development continues to be a process of refinement, with major changes occurring over a decade or more at a time. In short, the real issues don't change—just the marketing hype.

A quick glance at *Dr. Dobb's Journal* ten years ago underscores this. As with this month's issue, the April 1984 *DDJ* focused on cryptography, with C.E. Burton's two-part article entitled "RSA: A Public-Key Cryptography System." Without a doubt, cryptography is more important to a greater number of computer programmers and users now than it was a decade ago. Back then, RSA was still fairly new, and Burton's article was probably the first to bring RSA to the microcomputer platform. Today, RSA is clearly the dominant approach to cryptography. While better techniques may have been developed, it is unlikely that in the near future an alternative will achieve the same degree of commercial success as RSA.

1984 was also the year Judge Harold Greene became a household name, at least in the living rooms and kitchens of AT&T executives and stockholders. It was his consent decree, you'll recall, that led to the breakup of the most powerful telecommunications monopoly in the world. Now, ten years after, competition and innovation is finally beginning to catch fire with the Clinton administration's proposals to eliminate barriers between individual electronic-communication industries. However, proposed mergers between cable operators, Baby Bells, entertainment giants, online services, and the like may lead to electronic networks that dwarf the AT&T of yesteryear.

On the plus side, these mergers will fund the advanced information infrastructures we'll be using in the coming years. On the downside, the prospect looms that monopolistic megacorporations will be as unresponsive to the public well-being as Ma Bell was in the old days. For instance, Southwestern Bell, long the bellwether for the RBOCs when it comes to controversial legislation, is currently pushing for a law that would prevent the Missouri Public Service Commission from challenging phone company profits or rates. A draft of the bill reportedly says that Southwestern Bell "shall not under any circumstances be subject to any complaint or hearing as to the reasonableness of its rates, charges, rentals or earnings." Interestingly, this legislation was proposed on the eve of the telephone company's announcement that it had just achieved its best-ever fourth-quarter earnings. The company, which enjoys a monopoly, claims it needs the money to fund the construction of the information superhighway.

Clearly, both the federal and state governments have a responsibility to protect the public good against voracious proposals like that backed by Southwestern Bell. Likewise, the government needs to guarantee that proposed mergers won't result in a single company controlling both telephone lines and cable throughout an individual region. Competition and innovation have stood us well through the past ten years. They can get us through the next decade, too.


Jonathan Erickson
editor-in-chief

Develop Windows Applications Quickly and Easily

Phase3 Has Everything You Need

Visual Development

The Phase3 visual development environment provides a comprehensive suite of tools for screen creation. All standard Windows screen objects – push buttons, radio buttons, dialog boxes, etc. – are selectable from icon bars and are dynamically placed and sized on the screen as appropriate for the application. Standard Windows APIs are available from list boxes and are supported with on-line documentation.

Phase3 Database

Phase3 includes a relationally complete database supplied as a Windows DLL. The database supports complex data relationships and access and data manipulation by any language through a comprehensive suite of supplied database routines. Database integrity is enhanced with rollback recovery and transaction control. The Phase3 data dictionary simplifies data model definition and maintenance.

Query and Reporting

The Report Writer includes standard features like flexible headers, footers, free text, calculated fields, sort group sections and breaks, and subtotals. Reports can also include bitmaps of drawings and photographic images. Queries are executed with the Database Browser which also allows data entry and manipulation. Railway diagrams assist in query creation, prompting the user for appropriate selections and eliminating syntactical errors.

Lower CASE, E-R Modeling

Phase3 automates logical data model design with Entity-Relationship modeling. After a user describes entity relationships graphically and enters field descriptions, Phase3 generates the physical database based on an analysis of the entity relationships. Phase3 automatically includes foreign keys in appropriate tables and restructures the database as requirements change. Phase3 suggests appropriate referential integrity constraints to be enforced at runtime.

Hierarchy Chart

Phase3 maintains a graphical map of an entire application. The Hierarchy Chart includes all windows, dialogs, reports, and code routines. To access the underlying C or Pascal source code, simply point-and-click on any node. Phase3 generated source is easily accessed and extended with user written routines. User code is preserved even if an application is regenerated. The Hierarchy Chart and E-R Diagram provide instant core documentation for an application.

Help Generator

The Phase3 Help Generator allows the easy creation of a complete Windows application help subsystem including context sensitive on-line help. The Help Generator includes its own text editor that gives complete control over the content, appearance, and branching logic through highlighted trigger text. Phase3 generates a Windows compatible file with an ".HLP" extension that is then accessible from within the Phase3 created application. No other external tools are required.

"... a thoroughly remarkable product...
very impressive"

Jeff Duntemann
PC Techniques

"I was very impressed with Phase3, and using it made me wish I had learned Windows programming with a tool like this. Now that I know what it has to offer, I'll probably use it to build the frame for most of my programs."

L. John Ribar
Windows Tech Journal

"Phase3 hides the complexity of Windows programming but still allows an experienced programmer to drill down and extend generated C or Pascal source code, providing ultimate control."

Randy Goodhew
Computer Software Columnist

"Phase3 takes an intuitive, innovative approach to developing Windows applications. It's a pleasure to work with and has greatly boosted our productivity. One of the most important issues for us is that their technical support is excellent."

Reuben Halevi
ISOFT D&M

"It's easy to put together an application with Phase3 – everything's integrated. And the Phase3 database is terrific. It's closer to true relational than anything we've found."

Michael Erickson
Prism Business Solutions

CIRCLE NO. 825 ON READER SERVICE CARD



Royalty-Free Applications

Windows is a trademark of Microsoft Corporation. Turbo Pascal for Windows and Borland Pascal 7.0 are trademarks of Borland International, Inc. All other trademarks or service marks are recognized as the property of their respective owners.

Order Now
800-851-5650
Or Call for Free Demo Disk
Fax (805) 641-9083

CALL NOW FOR COMPETITIVE UPGRADE PRICING



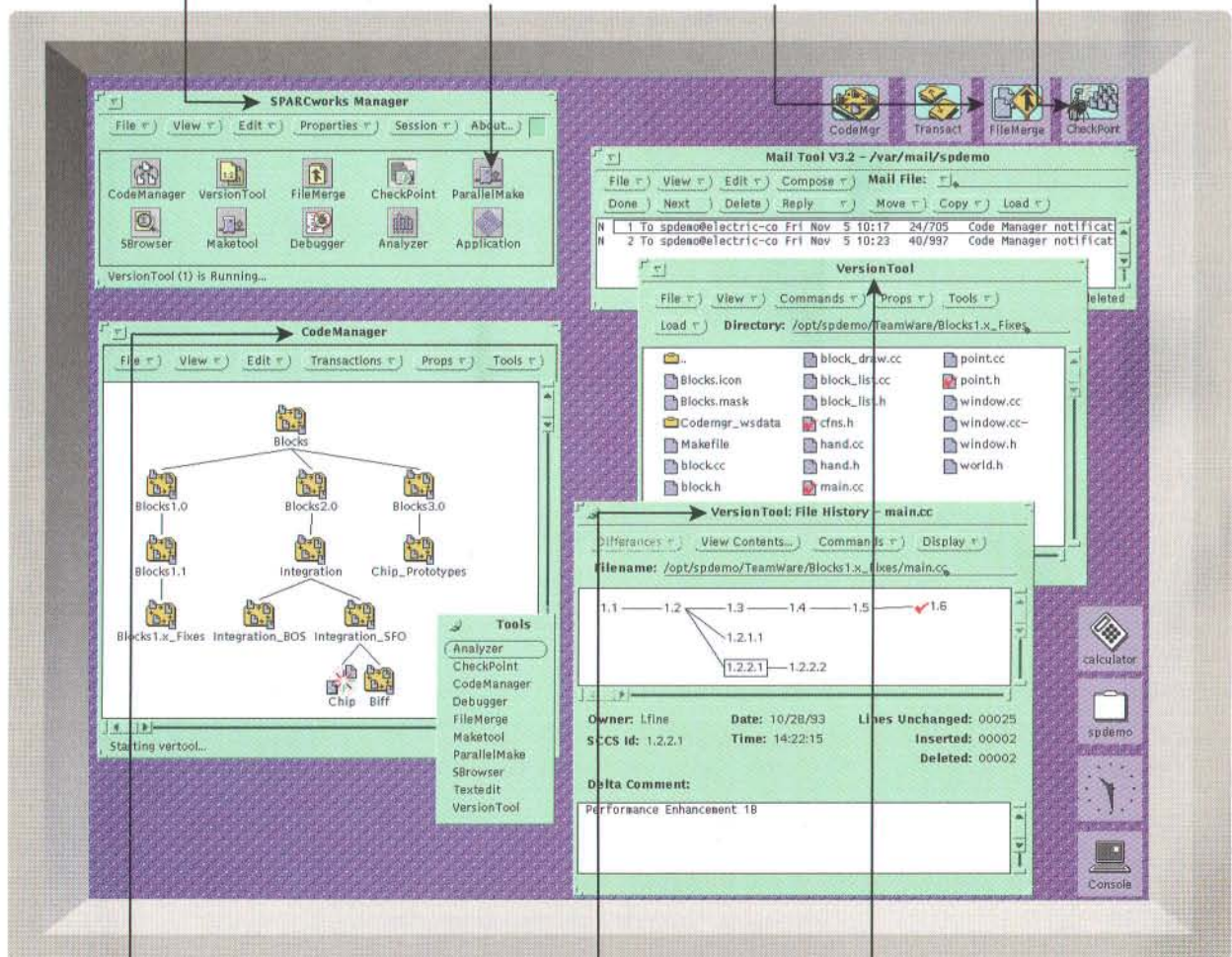
Can you really look us in the face and say you don't need better code management?

SPARCworks Manager—
Coordinate your development sessions and tools. User extensible.

Parallel Make—
Dramatically accelerate project builds by leveraging the power of your compute server.

FileMerge—
Graphically compare and merge source code. Automatically merge concurrently modified files.

CheckPoint—
Capture complete project releases for later retrieval.



Code Manager—
Easily organize and integrate work from multiple developers, sites, and platforms. Automatically track and inform of conflicting source code changes.

Compatible—
Begin using without any special preparation or administrative requirements.

Version Tool—
Accelerate version control. Graphically view source file history as well as concurrent modifications.

Because this is the new face of software development—integrated, organized, productive, and very, very easy to use.

It's called SPARCworks™/TeamWare for the Solaris® operating environment.

And it's from SunPro™, the software development arm of Sun Microsystems, Inc.®

What makes our SPARCworks/TeamWare products so great? Let us answer that question with a few of our own.

How would you like to have multiple developers working on the same source base without getting in each other's way?

Or how about doing source code development, quality assurance and testing, and release engineering at the same time?

Regardless of whether they're in the same country, or even on the same network?

And manage more than one release at a time, with code coming from different locations?

Perhaps building project components in parallel and integrating them later appeals to you?

We thought so.

You see, SPARCworks/TeamWare can do all that and more. Because it was specifically designed to solve the problems of today's complex, multi-developer, multi-site, multi-platform development projects.

No matter how big or small those projects are. SPARCworks/TeamWare works with the SPARCworks development environment to be totally scalable and extensible—from small groups to large ones, across geographically distributed sites, and disparate platforms.

It's also customizable to each developer's or team's way of working, right down to specific operations.

Which brings us to the next point.

SPARCworks/TeamWare builds on standards your developers already know and use: the UNIX® SCCS utilities, NFS®, and X window system utilities. So there's no ramp-up time, no new system administration, no conversion, and no special databases.

You're ready to run as soon as you load it.

What's more, because SPARCworks/TeamWare uses a graphical approach, developers can work in parallel, and actually see what they—and everybody else—are doing, all at the same time.

And what would you expect to pay for all this?

A lot less than you'd guess.

Under \$1,000 a seat.

Call us at 1-800-2SUNPRO for a copy of the SPARCworks/TeamWare Solutions Guide including real life success stories from SPARCworks/TeamWare users, or our 30-day Try and Buy CD which lets you try SPARCworks/TeamWare before you buy it.

So why choose between crude version control tools on one end or overly complex (and expensive) configuration management systems on the other?

Especially now that you've come face to face with something better.

 **SunPro**
A Sun Microsystems, Inc. Business



Without SPARCworks/TeamWare, everybody on your team might as well be locked in their own rooms.





Pairing C and C++

Dear *DDJ*,

In his article "Programming Language Guessing Games" (*DDJ*, October 1993), P.J. Plauger expresses confusion over the popularity of C++ since it is so complex and presents an exceedingly complex algorithm for pairing teams in a round-robin tournament.

Yes, C++ is a complex language. Plauger is right in suggesting we choose and stick with a subset of the language. C++ is like English. You can use it to state something in a very complex fashion, or very simply. The latter is usually the stronger statement. The round-robin tournament problem illustrates this very well.

The physical-education community has developed a simple algorithm for pairing teams. First, list the teams on pieces of paper. Leave one piece blank, if necessary, to ensure an even number of teams. Lay out the papers in two rows; that's day #1. Hold the upper-left piece in place, and rotate all the other pieces of paper; you then have day #2. Repeat the process for each succeeding day until you are back at your original positions. For six teams, the rotation would be as in Figure 1.

The C++ program implementing this algorithm is equally straightforward. Note the simple, elegant power of the *for* statement controlling the inner print loop, which I show in Figure 2.

Jay Frederick Ransom
Oxnard, California

Day 1:	1	2	3
	6	5	4
Day 2:	1	6	2
	5	4	3
Day 3:	1	5	6
	4	3	2
Day 4:	1	4	5
	3	2	6
Day 5:	1	3	4
	2	6	5
Day 6:	Repeats Day 1		

Figure 1

Keep It Simple

Dear *DDJ*,

I was glad to read Michael Swaine's "Programming Paradigms" (*DDJ*, November 1993), which gives Forth a plug, even in a lighthearted way. It's a far cry, though, from the good old days of a decade ago when *DDJ* annually had an entire issue devoted to Forth.

Part of the Forth Standards efforts are confounded because the creator of Forth, Charles Moore, doesn't believe in standards for Forth. Moore considers Forth to be a program-development environment that increases productivity by speeding up the programming cycle. To keep it simple, small, and speedy (KISS), certain design decisions resulted in using postfix notation, threaded code for compiling, a dictionary to hold functions, separate stacks for data and return addresses, and a (usually emulated) stack-based processor. As a result, the Forth language is an outgrowth of the Forth system, rather than a construct in its own right.

Moore works mostly with embedded systems, and varies the basic Forth to match the hardware and program-design requirements. Forth then behaves like assembler. A few primitive words (func-

tions, opcodes) are used to extend Forth to develop the data types and structures that particular program requires.

On the other hand, a large group of programmers want to use Forth in symbolic programs: word processors, spreadsheets, graphics programs. They believe a Forth language without a standard is Forth in chaos. Many come from a traditional-language background—Fortran, Pascal, C, and the like. They want Forth to look more like the language they are familiar with, so they push for CASE statements, local variables, more stacks, string functions, floating-point numbers, graphics functions, and so on, as part of the standard Forth. It comes to this: Do you want Forth on a floppy or a CD-ROM?

Walter J. Rottenkolber

Mariposa, California

Windows Setup Follow Up

Dear *DDJ*,

I wish that you had published Walter Oney's article, "Examining the Windows Setup Toolkit" (*DDJ*, February 1994) a few months earlier. Last November, I had the pleasure of building a setup program for an in-house software package using the setup toolkit from the Microsoft SDK.

```
#include <iostream.h>
#include <string.h>

char teams [100] [50],      // 100 teams ought to cover the field
                                // long names not allowed
    temp [50];
int  n_teams = 0,
    first_half,
    second_half,
    day,
    rotate;
char last_name;

main ()
{
    // Read in team names
    do
    {
        cin >> teams [n_teams];
        last_name = teams [n_teams] [0];
        n_teams++;
    } while (last_name != '*'); //Delete the * team:
    n_teams -= n_teams % 2;    //Another simple, yet strong, expression

    // print out playing schedule
    for (day = 1; day < n_teams; day++)
    {
        cout << "\n\nDay " << day << "\n\n";
        for (first_half = 0, second_half = n_teams - 1;
            first_half < second_half;
            first_half++, second_half--) //What you can do with a for loop!
            cout << " "
                << teams [first_half]
                << " vs "
                << teams [second_half]
                << "\n";

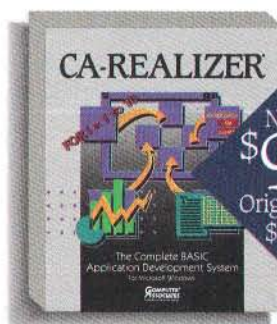
        // Rotate teams for next day

        strcpy (temp, teams [1]);
        for (rotate = 2; rotate < n_teams; rotate++)
            strcpy (teams [rotate - 1], teams [rotate]);
        strcpy (teams [n_teams - 1], temp);
    }
    return 0;
}
```

Figure 2

A Better Basic.

New CA-REALIZER® 2.0 Beats Visual Basic 3.0.

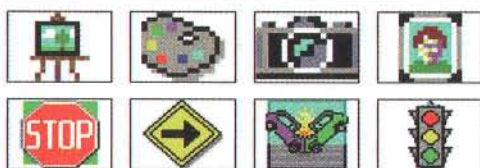
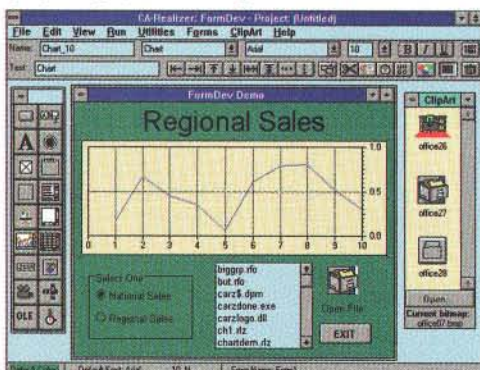


Compared to Visual Basic 3.0, CA-REALIZER® 2.0 is half the cost and twice the product.

CA-REALIZER is already the easiest, most powerful BASIC for Windows and OS/2, and with version 2.0 it's even better. No one can match our combination of features, ease of use, and price. No one can make it easier to port your applications from QuickBASIC, and

no one can make it more fun to develop for Windows and OS/2.

CA-REALIZER also comes with a huge array of powerful, plug-and-play tools like spreadsheets, charts, text editors, animation,



graphics tablets and database forms, along with many features other BASICs don't offer. Arrays are re-dimensioned and processed automatically. Algorithms can be written as formulae instead of complex looped expressions. Once an application is complete, compile it into a stand-alone OS/2 or Windows

application and distribute it royalty-free with the run-time module that's included. And you can generate an installation disk with the push of a button.

PC Computing said, "No other Windows BASIC can match it for power and breadth of features."

At \$99, no one can match our price either, and it includes our award-winning Windows report writer absolutely FREE.



Only CA-REALIZER Gives You All Of This:

- Develop for Windows and OS/2
- Create and save libraries of pre-coded and formatted controls dynamically from within REALIZER
- Use any standard Windows and OS/2 custom control in FormDev
- Integrated Programmable Application Tools such as Charts, Spreadsheets, Text Editors, Graphics, Animation, and a Scheduler
- Easy migration of QuickBASIC applications to Windows and OS/2
- Import/Export of 1-2-3, Excel, CA-Supercalc, CA-Compete! & Xbase Files
- Interactive WYSIWYG application design
- Generation of user accessible & modifiable BASIC code
- Full support for DDE and DLLs
- Instant database forms from DBF files
- Dynamically expandable multi-dimensional arrays with a full range of array and matrix operators and functions



Call 1-800-225-5224, Dept. 25302
Today Or See Your Local Dealer.

This offer is good for a limited time only. Call today and find out how much better BASIC can be.

COMPUTER ASSOCIATES
Software superior by design.

New CA-REALIZER 2.0

(continued from page 10)

I heartily agree with Walter: The setup toolkit is a credible toolkit (and free for SDK owners), but its documentation leaves you wishing for more.

There is one point in Walter's article which I can simplify. He recommends hand-modifying the .INF file produced by the DSKLAYT program. DSKLAYT is used to lay out the files to best fit on the setup disks. This program requires that you specify all of the files that will reside on the setup disks. This includes not only your applications files but the files that control the setup process. The .INF file, which is generated by the DSKLAYT program, controls the copy-

ing of the files from the setup disks to the user's hard disk. Normally, all of the setup files will also be listed in the .INF file and copied to the user's hard disk. Walter suggests removing these files from the .INF file by hand.

A better approach is to specify that the setup files be placed in a different section of the .INF file. This is controlled by the DSKLAYT program by filling in the "Put In Section" entry for each of the setup files. When this field is left blank, the files are put into the default section named "Files." I specify that all the setup files be placed into the section *SetupFiles*. When your setup script runs, it calls the function *AddSection-*

FilesToCopyList and specifies the section name to add. For simple installation, only one call specifies the section "Files."

This is also the procedure to use when you want to allow the user to selectively setup portions of your application. You group your files in sections and allow the user to select which sections to setup. It is then a simple matter to call *AddSectionFilesToCopyList* for each section that is to be setup.

Gene Psoter
Atascadero, California

Random Thoughts on the Stock Market

Dear DDJ,

Tom Swan's "Algorithm Alley" column (DDJ, December 1993) correctly points out that a group of numbers, which are alleged to be random, must satisfy a lot of tests. But the highlighted example—the stock market—is not a good random sequence. The market is somewhat unpredictable. But successive prices are very strongly correlated. The distribution of the first differences (daily changes) is very uneven with far too many very small changes. You'd quickly scrap a random-number generator that created numbers like that. The other example, lottery numbers, is fine.

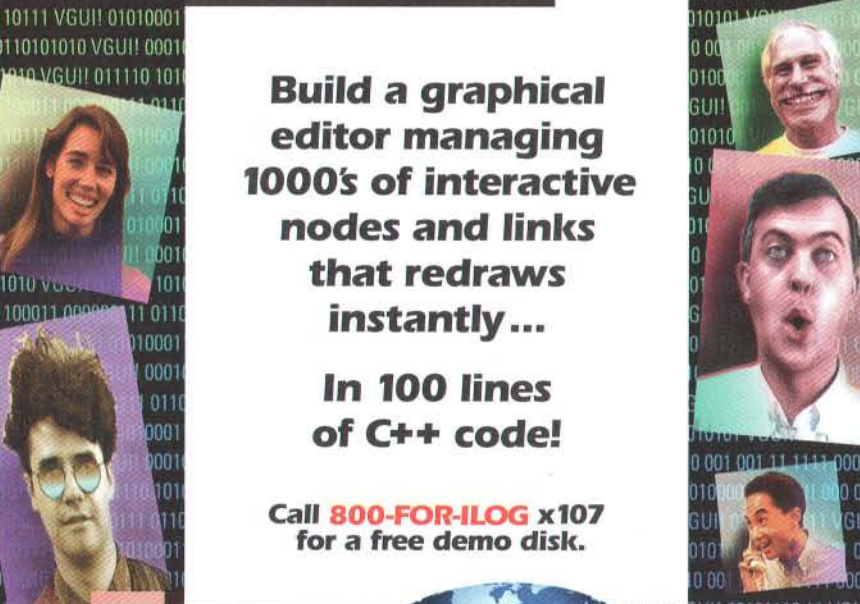
Donald Kenney
CompuServe 72630,3560

Tom replies: Thanks for your letter, Donald. You're right that successive stock-market prices are very strongly correlated, but so are the sequences produced by common random-number generators. In fact, so-called "random numbers" are completely predictable to produce the same sequence—just re-run the program using any value in the sequence as the starting seed! Stock market prices are more random because they are truly unpredictable. If that were not so, as I stated in the article, everyone would be wealthy.

I understand your point that stock-market prices themselves would not be suitable as direct substitutes for common random-function output, but I never said they were. If you use the Dow Jones Industrial Average to program a lunar-lander simulator, the landing module may crash. (Let's hope the stock market doesn't.) Seriously, I am not suggesting using stock prices as random numbers; only that the behavior of the stock market is an example of true randomness. Random-number generators are misnamed because their output is predictable, and therefore, not actually random. A generator's output may appear to satisfy some conditions of randomness, but only real-world events are truly chaotic.

(continued on page 16)

ILOG VIEWS™



Build a graphical editor managing 1000's of interactive nodes and links that redraws instantly ...

In 100 lines of C++ code!


Call 800-FOR-ILOG x107 for a free demo disk.

ILOG VIEWS enables the development of Very Graphical User Interfaces (VGUI™).

ILOG VIEWS is a portable C++ library of graphic and interactor objects that extends standard GUI environments with real graphics.


ILOG VIEWS lets you draw, objectize, edit and customize. It gives instant access to a comprehensive library of structured graphical objects and interactors (select, move, resize, zoom) as well as PowerObjects (network, charts, spreadsheet, Gantt chart, hypertext, client-server).

Graphics are separate from interaction—making it possible to shrink application code by 1:20 while boosting performance!



iLOG

ILOG, Inc.
2073 Landings Drive
Mountain View,
CA 94043 USA
phone: (800) 367-4564
(415) 390-9000
fax: (415) 390-0946
email: vgui@ilog.com



CIRCLE NO. 913 ON READER SERVICE CARD

MAKE YOUR CODE FLY



When you're pressing the limits of software performance, every microsecond counts. Veering from the optimum compilation course increases execution time.

That's why professional C/C++ developers use High C/C++. MetaWare's High C/C++ compilers offer eight levels of global optimization, plus specific optimizations for particular processors. The executables consistently out-perform competitors in benchmark testing.

Besides offering rapid execution time, MetaWare's High C/C++ compilers provide a myriad of controls that allow you to customize and fine-tune code generation. There are command-line toggles and pragmas for customizing your application.

The compilers generate informative error and warning messages, and comply with major industry standards.

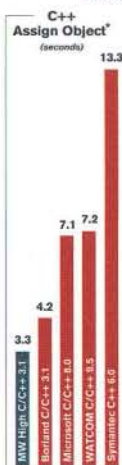
For years, MetaWare has supplied major OEMs

such as AMD, AT&T,

CenterLine, IBM, Intel, and NCR, with 32-bit compilers for a variety of different processors and platforms. These OEMs depend on MetaWare for superior products and support.

Call (408) 429-6382, or email: techsales@metaware.com, for more information about features and specific platforms.

Dhrystone 2.1* Dhrystones/sec.	
MW High C 3.2	53,191.5
Sun Compiler	44,642.9
Whetstone* KWhets/sec. (single precision)	
MW High C 3.2	22,727
Sun Compiler	19,230



High C/C++ for OS/2, NT, DOS, Windows, AIX, UNIX SVR4, and Solaris

*Benchmark source and other performance information available upon request. Dhrystone benchmarks use the -O option. MetaWare, the MetaWare logo, High C, and Professional Pascal, are registered trademarks, and High C/C++ and High C++ are trademarks, of MetaWare Incorporated. Other names are trademarks of their respective companies. Copyright 1994 MetaWare Incorporated, 2161 Delaware Avenue, Santa Cruz, CA 95060-5706.

CIRCLE NO. 158 ON READER SERVICE CARD



Programmer's Paradise®

C++/Views 3.0 by Liant Software Corp.

The best object-oriented application framework for developing multi-platform, native GUI programs using C++. Includes a library of 100+ C++ classes that solve a broad range of software development problems including interface design, data management, event processing, & more. Includes C++/Views Constructor, a unique development tool that lets you work visually with the C++/Views class library. The Constructor unites an Interface Builder for creating portable resources & a Browser so you can switch from drawing & archiving your portable resources to editing the code that calls these resources. No royalties or RT fees. Source code is free.



List: \$749 Special Ours: \$565 FAXcetera #: 1952-0021



WindowsMAKER Professional 5.5 by Blue Sky Software

NEW VERSION!

If you're serious about C/C++ Windows development, WindowsMAKER™ Professional 5.5 is the most powerful Prototyper and C/C++ Code Generator for Windows, Win32 and Windows NT. This award-winning product offers more functionality & ease-of-use than any other tool. Create full-featured Windows Applications: MDI, Toolbars, Status bars, Templates, On-line Help, Graphical 3D buttons, Edit During Preview & much more. TrueCode™ technology ensures that user code is 100% preserved during code regeneration. Supports ANSI C, MFC C++, OWL C++ & more. Uses Switch-IT™ Code Generation Modules for generating code for specific platforms, allowing migration between languages, C++ libraries & platforms. Highly recommended!

List: \$995 Ours: \$875 FAXcetera #: 2602-0003

c-tree Plus® by FairCom

DOS • WINDOWS • NT • UNIX • OS/2 • SUN • RS6000 • HP9000 • MAC • QNX • BANYAN • SCO. This well known, highly portable data management package has become established as the tool of choice for commercial development. Offering unprecedented data control, programmers may choose from direct low level access, ISAM level convenience, or SQL access with the FairCom Server. Single User, MultiUser, or Client/Server, ANSI Standard.



List: \$595 Ours: \$505 FAXcetera #: 1381-0008
Call Programmer's Paradise Italia for special pricing in Europe.

Janus/Ada 9X Professional Development System

by R.R. Software

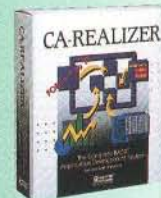
New!
Ada 9X for the PC

The leader in PC-based Ada, proudly introduces the first comprehensive Ada 9X programming systems. Ada 9X brings the latest in Object-Oriented programming to Ada, and you can get it today complete with numerous professional tools. Special introductory pricing is good only until July 31, 1994.

MS Windows NT or 32-Bit DOS Extender:

List: \$995 Ours: \$625 FAXcetera #: 1876-0001

CA-REALIZER for Windows & OS/2 by Computer Associates



Defines a new generation of development tools that handles the mechanics of event-driven programming, message passing, process sharing and other complexities behind the scenes. Combines a structured superset of BASIC extended to access Windows and OS/2 objects and resources, a visual development tool and Programmable Application Tools. CA-Realizer will help you create spreadsheets, charts, text editors, animation, graphics tablets and user-friendly forms from tools that can be created and manipulated by simple commands.

List: \$295 Ours: \$79 FAXcetera #: 1004-0008

PRODUCT OF THE MONTH

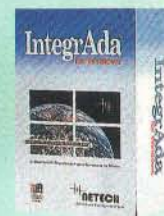
Borland C++ 4.0 by Borland International



New Borland C++ 4.0 is visual, and that's just the beginning. Only Borland C++ has true C++ support with exceptions, templates and the easiest, most powerful suite of object-oriented tools available today. With full support for DOS, Windows, Win32s and NT, it's the C++ development system you've got to have.

	List: \$499	Ours: \$329
Upgrade	List: \$199	Ours: \$189
FAXcetera #: 1861-0016		

IntegrAda for Windows by Aetech Inc.



Complete Ada Programming Support Environment (APSE) for Windows, adding a language sensitive editor; automatic code generation & error checking; interactive error correction; pop-up menus to completely customize the environment to the standard Windows features. Allows you to quickly code, correct, compile, & link your Windows programs using Ada. Includes environment, libraries, validated Ada compiler, Ada Windows Libraries, Header Files, Resource & Help compilers, & MS Linker. (ACVC 1.11 validation pending). No RT Royalties!

List: \$995 Ours: \$895 FAXcetera #: 2358-0003



Symantec C++ Professional 6.1 by Symantec

The new version 6.1 not only enhances product stability and reliability but also brings new features that 6.0 customers asked for. Now you get: full template debugging; improved hierarchical project manager; customizable color-syntax highlighting; & enhanced 32-bit support with 32-bit MFC 2.0 on the CD-ROM. So don't wait. Try Symantec C++ 6.1 and find out why the critics are raving about this new breakthrough in programming systems.

Competitive Upgrade for Borland or Microsoft customers \$189.

	List: \$499	Ours: \$299
Comp. Upg.:	List: \$199	Ours: \$189
FAXcetera #: 2132-0038		

9-788-444-8008

Programmer's Paradise®
Italia
Phone: 39-2-480-16053
FAX: 39-2-480-16039

Best Sellers

Call for a
FREE
Catalog!



Network C Library by Automation Software Consultants, Inc.

NEW
VERSION
3.0!

The most comprehensive library available for NetWare software development, supporting all versions of advanced NetWare. Over 450 C functions, include any features from the NetWare command line utilities and menu utilities in your C or BASIC programs for Windows or DOS. No licensing hassles. C library source code available.

List: \$395 Ours: \$355 FAX_{cetera} #: 1004-9201



Microsoft Visual C++ Development System v1.5 by Microsoft Corporation

Master the power of OLE 2 & the database access flexibility of ODBC with MS Visual C++ v1.5, Prof. Edition. V1.5 allows you to create high-performance database apps & powerful reusable OLE 2 components. Includes Visual

Workbench, App Studio, and MS Foundation Class Library 2.5.

MS Visual C++ 1.5	List: \$599	Ours: \$379
w/Chinon CD-ROM Drive	List: \$749	Ours: \$699
1.5 Upgrade	List: \$ 99	Ours: \$ 89
w/Chinon CD-ROM Drive	List: \$519	Ours: \$479
Visual C++ 16/32 bit Suite	List: \$199	Ours: \$185
FAX_{cetera} #: 1269-0056		

WATCOM FORTRAN 77³² Version 9.5 by WATCOM

32-bit optimizing FORTRAN 77 compiler and tools for DOS, Novell NLM, OS/2 2.x, Windows NT, Win32s, & Windows 3.x. Comprehensive language support with FORTRAN 90, DEC VAX & IBM VS language extensions. Advanced processor optimizations including 486 & Pentium instruction scheduling. Multi-platform toolset includes linker, debugger, profiler, royalty-free DOS extender with VMM & more.



List: \$599 Ours: \$349 FAX_{cetera} #: 1683-0005

RoboHELP® 2.6 by Blue Sky Software

RoboHELP® 2.6, the best-selling Help Authoring Tool for Windows & Windows NT, offers full document to Help system conversion & vice versa. Turns Word for Windows into a fully functional hypertext authoring system capable of producing Windows Help files as easily as it does plain text. Fill in the actual Help text when prompted. RoboHELP takes care of generating the RTF, HPJ & H files. Link tester allows you to simulate your design before you compile. Full support of Word 2.0 & Word 6.0, & all features in the Windows Help Engine, such as macros, secondary windows, & multiple hotspot graphics.



List: \$499 Ours: \$439 FAX_{cetera} #: 2602-0005

NEW THIS MONTH

ReferencePoint Personal Assistant by ReferencePoint

ReferencePoint Personal Assistant automatically makes an index of any information that you create or modify anywhere on your computer system, so that you can find any file, instantly, whether from a word processor, database or spreadsheet. So, type what you want to find, and Personal Assistant will locate every file in the computer containing those words. ReferencePoint Royalty Free SDK for DOS, Windows, Mac, OS/2 and UNIX also available from Programmer's Paradise®. Call for pricing.



List: \$99 Ours: \$ 79 FAX_{cetera} #: 1008-5001

WATCOM™ C/C++32 v9.5 by WATCOM

C/C++32 is a professional, multi-platform C and C++ development system supporting 32-bit extended DOS, OS/2 2.x, Windows 3.x, Windows NT, Win32s, and AutoCAD ADS/ADI. The complete toolset includes: C and C++ optimizing compilers, royalty-free DOS extender with VMM support, licensed components from the MS Windows 3.x SDK, interactive source-level debugger, linker, profiler, supervisor for executing 32-bit applications and DLLs under Windows 3.x, 32-bit run-time libraries for extended DOS, OS/2 2.x, Windows 3.x and Windows NT, and more.



List: \$599 Ours: \$349 FAX_{cetera} #: 1683-0003



Mwave™ Developers Toolkit with IBM WindSurfer™ Bundle by Intermetrics, Inc.

What once was multiple products and toolsets is now bundled in one multimedia PC add-in card and Toolkit. The Mwave WindSurfer Communications Adapter is a "works out of the box" data/FAX modem, sound, voice messaging and telephone answering card with application software. It is bundled with the Mwave Developers Toolkit so you can build software that takes advantage of the Mwave digital signal processing (DSP) platform that drives WindSurfer. Try out the magic of a software upgradeable and programmable Mwave PC solution. You'll be developing for the future and using it today! And you can do it for an incredibly low price!

List: \$900 Ours: \$495 FAX_{cetera} #: 1012-4601

GUARANTEED BEST PRICES! (Call for Details)

To order call: 800-445-7899
Corporate (CORSOFT): 800 422-6507
FAX: 908 389-9227
International: 908 389-9228
Customer Service: 908 389-9229
Programmer's Paradise Italia:
39-2-480-16053

For more information on the products featured on these pages call
FAX_{cetera}®: (201) 762-1378

Programmer's Paradise®
1163 Shrewsbury Avenue
Shrewsbury, NJ 07702

* All prices are subject to change without notice.
Call for details on return policy and shipping charge.

CIRCLE NO. 195 ON READER SERVICE CARD

8
0
0
4
4
5
-
7
8
9
9


```

procedure GetMaxC(H, V : Double;
                  var C : Double);
[ Get Max C for given H and V ]
const
  Pi      = 3.14159265358979324;
  Pi2o3   = 2.0 * Pi / 3.0;
  DpR     = 180.0 / Pi; { Degrees per
                        Radian }
var
  HRad : Double;
  C1, C2, C3 : Double;
begin
  HRad := H / DpR;
  C1 := Cos( HRad + Pi2o3 );
  C2 := Cos( HRad );
  C3 := Cos( HRad - Pi2o3 );
  if (C1 > 0.0) then
    C1 := V / C1
  else
    if (C1 < 0.0) then
      C1 := (V - 1.0) / C1
    else
      C1 := 1.0;
  if (C2 > 0.0) then
    C2 := V / C2
  else
    if (C2 < 0.0) then
      C2 := (V - 1.0) / C2
    else
      C2 := 1.0;
  if (C3 > 0.0) then
    C3 := V / C3
  else
    if (C3 < 0.0) then
      C3 := (V - 1.0) / C3
    else
      C3 := 1.0;
  C := C1;
  if (C2 < C) then C := C2;
  if (C3 < C) then C := C3;
  if ((V <= 0.0) or (V >= 1.0))
    then C := 0.0;
end; [ GetMaxC ]

```

Figure 3

(continued from page 12)

Putting HVC Back in Order

Dear DDJ,

In Maxwell T. Sanford II's letter (DDJ, November 1993) about my sidebar, "Putting Colors in Order" (DDJ, July 1993), Maxwell was too restrictive in assigning $C_{max} = \min(V, 1 - V)$. It is true that R, G, and B must be in the range (0.0, 1.0), but the Chroma C can be as large as 2/3. This is true, for example, when $V = 2/3$ and $H = 0$. My new paradigm for representing a color is a color bubble, not a color cone. Given values for H and V, the max C is given by the procedure *GetMaxC* in Figure 3.

Harry J. Smith

Saratoga, California

Processor Scenarios

Dear DDJ,

Even after many months, I found Nick Tredennick's article, "Computer Science and the Microprocessor" (DDJ, June 1993) very interesting. His market model and definitions make a lot of (common) sense. During the course of the article, there is a recurrent theme of RISC vs. CISC and PC software vs. workstation software, and it becomes apparent that Nick feels that the two will never mix.

However, emerging software technology is providing the inevitability that the two worlds will mix. Without going into its relative merits, Microsoft's Windows NT is one of the first to desegregate hardware systems. NT is already on several microprocessors, and the list is likely to grow. This can only serve to help the microprocessor market waters find their own level. It should mean that the largest market share (CISC) will drop some, and the lowest market share (RISC) will rise.

The ultimate scenario is that any microprocessor house can provide a microkernel driver for their product and "Intel Inside" will have no more meaning than "GE Inside" for toaster-heating elements (although I've heard GE is considering using the Pentium in its toasters as heater elements). Then we can all pick our own favorite processor for the job at hand. Aaron Goldberg, in the September 27, 1993 *PC Week*, called this the "Esperanto of Tomorrow" (and showed why a multiplatform operating system has value, whereas Esperanto failed for lack of interest).

Jonathan Platt

Pipersville, Pennsylvania

DDJ

THE TOP 20 REASONS TO USE PAIGE IN YOUR 1994 PRODUCT LINE.

PAIGE™ is a cross-platform text processing/page layout engine. Here's why you should own it.

2 Open-Architecture Design.

PAIGE is extremely flexible and "forgiving". It makes no assumptions about the contents of a document—or even its shape and coordinates. You can control every aspect of text/graphic formatting, editing and display.

3 Stylized Text. Extensive stylization and text formatting—with no memory limits. Features include paragraph formatting, kerning, hidden text and variable tab stops.

4 Embedded Objects. Embed objects or data elements into the text stream—graphics, video, etc.

5 Containers & Shapes

Document data will adhere to and wrap inside or outside application-defined columns, linked containers and irregular shapes.

6–20 Hypertext capable

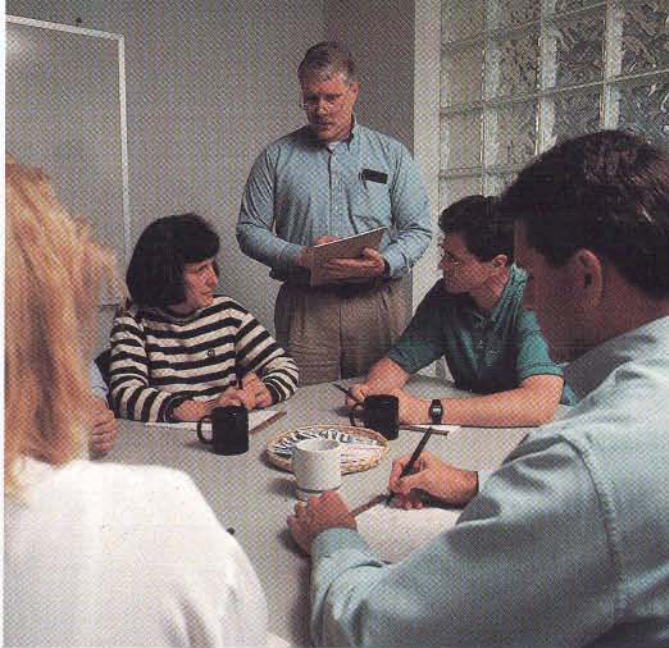
• Advanced word processing features • Multi-level "Undo support" • Print View Scaling • Written in portable C code (object or Source) • Virtual Memory • Uses no global variables • Supports most compilers, development systems & class libraries • World Script compatible • Mail-merge • High-level scrolling and pagination • Style sheet support • Discontinuous Highlighting • One year Tech Support • ROYALTY FREE

WINDOWS, Macintosh, PowerPC Versions Available

PAIGE is a multi-platform development engine. It allows you to design the majority of your application in "CPU independent" code. Each additional platform version requires you to change only the platform specific portions of the code while maintaining data and application compatibility.

Join the hundreds of major software publishers planning to use **PAIGE** as their development tool of choice. For complete technical/pricing summaries contact **DataPak Software** at **800-327-6703** or **(206) 573-9155**.

1. 1995's Bottomline...



"We chose PVCS because the vendors of our new development tools recommended it. They told us Software Configuration Management was a necessity for productive development on the LAN. I'm glad we listened to them. PVCS has proven to be invaluable. We now have superb control over all development efforts."

The Major Development Tool Vendors Agree: PVCS Is Essential for Productive Development on the LAN

Selected Vendors:

Borland International IDE
 Digitalk Team/V
 Gupta Technologies, Inc.
 HP SoftBench
 IBM SDE/6000
 IBM OS/2
 Interactive Unix
 Micro Focus COBOL
 Microsoft Visual C++
 Microsoft PWB
 Powersoft PowerBuilder
 SCO
 SunOS
 SunSoft
 Solaris
 Symantec C++ IDDE
 And many more...



Leverage Your Tool Investment with PVCS

There's a reason why all major Client/Server and LAN development vendors have formed alliances with INTERSOLV. It's PVCS—the standard for Software Configuration Management on the LAN. It's a vital ingredient for productive LAN development.

Every Major Development Tool Vendor Partners with Intersolv

The list on the left is only partial. PVCS is available in all the environments developers work in—today and tomorrow. That means less training—you don't have to learn another SCM tool if there's a change. That means more efficient development and easily enforced standards. It means consistent audit trails for all development, and the ability to correct errors and anomalies at any time.

Serious Application Development Demands PVCS

The trend is clear. Important applications in your company are moving to the LAN. They need the control and reliability that PVCS provides. Your development tool vendors know that—that's why they recommend PVCS.



**Call Today for
 More Information:
 800-547-PVCS Ext. 20**



The Standard for Software Configuration Management on the LAN

Inside U.S.A. • 1700 NW 167th Place • Beaverton, OR 97006 • 503-645-1150 • FAX 503-645-4576 • E-Mail PVCSINFO@INTER SOLV.COM
 Outside U.S.A. • Abbey View • Everard Close • St. Albans • Herts AL1 2PS • United Kingdom • Tel 0727 812 812 • FAX +44 727 869 804

CIRCLE NO. 190 ON READER SERVICE CARD

The Cambridge Algorithms Workshop

Designing fast, secure algorithms for high-performance applications

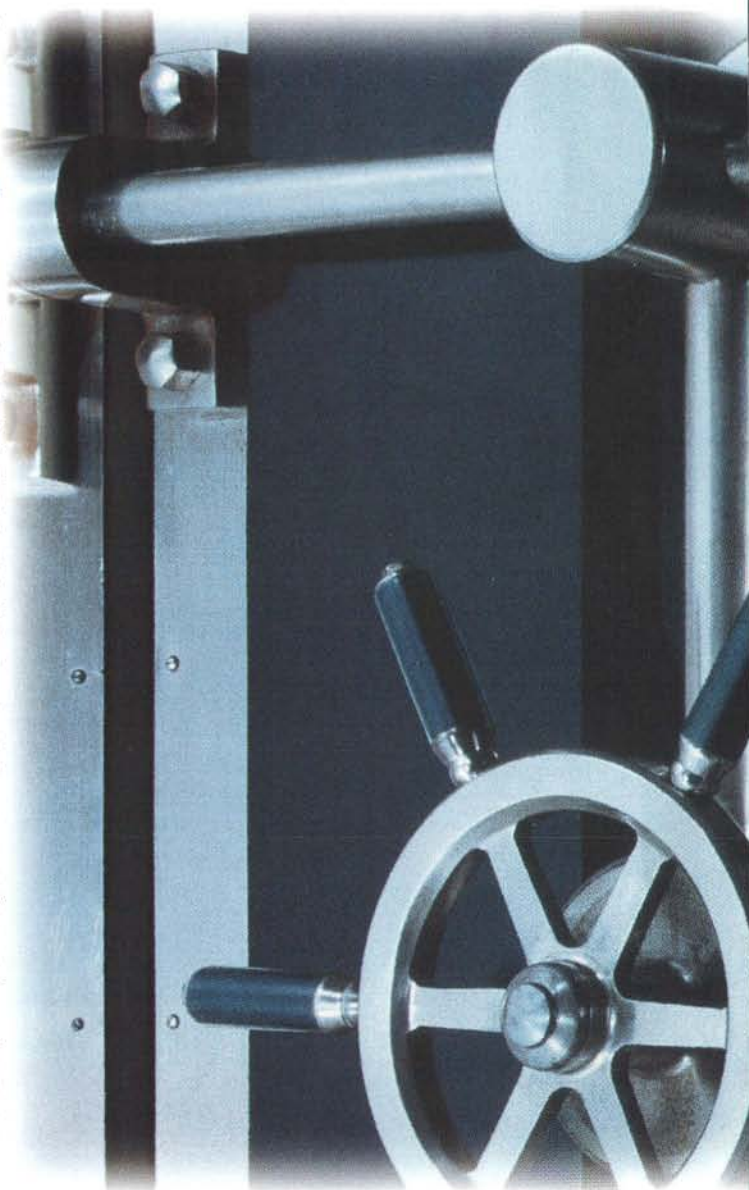
Bruce Schneier

In December 1993, the Cambridge Algorithms Workshop, hosted by the Cambridge University Computer Laboratory, brought together leading figures in the field of encryption who presented, examined, and challenged new encryption algorithms designed to run quickly in software. The reasoning behind the focus on encryption is that constructing algorithms which are both fast and secure is the core problem of classical cryptology. However, recent developments, such as differential attacks on block ciphers and correlation attacks on stream ciphers, have forced cryptanalysts to rethink classic encryption algorithms such as those in Table 1. At the same time, the need for fast, efficient, and safe encryption at the application level has increased—DES (even triple-DES) may be fast enough for e-mail, but it's too slow for emerging high-bandwidth applications.

The goal of the conference, therefore, was to propose new algorithms capable of encrypting data at a dozen or so clock cycles per byte for the emerging class of high data-rate applications.

In this article, I'll cover the conference highlights and examine the current state of encryption technology in general. For specifics on the workshop, the proceedings will be published later this year by Springer-Verlag as part of their *Lecture Notes in Computer Science* series. (Call 201-348-4033 for availability.) Additionally, many of the topics and algorithms touched upon in this article are discussed in my book, *Applied Cryptography* (John Wiley & Sons, 1994).

Bruce, an independent software developer and author of Applied Cryptography: Protocols, Algorithms, and Source Code in C (John Wiley & Sons, 1994), presented a paper at the Cambridge workshop. Bruce can be contacted at 708-524-9461 or schneier@cbinet.com.



The Algorithms

In all, ten complete algorithms were presented at the Cambridge Algorithms Workshop, all secret-key, not public-key algorithms like Diffie-Hellman, RSA, and the U.S. Government's Digital Signature Standard (DSS). (For more on public keys, see "Untangling Public-Key Cryptography," *DDJ*, May 1992.)

Jim Massey presented SAFER, a 64-bit block cipher with a 64-bit key. It is designed for implementation on simple processors on smart cards and only uses addition and multiplication mod 256, and exponentiation and logarithms in the multiplicative group of GF(257). Although Massey originally developed SAFER for Cylink Inc., the company has decided to place this algorithm in the public domain in the hope that it will become the new standard for software encryption. This algorithm will probably be important. In fact, former Soviet cryptanalysts in Yerevan, Armenia have been—without success—trying to break SAFER for over a year.

Matt Robshaw of RSA Data Security presented a fast block cipher based on the same principles as the MD5 one-way hash function (see my article, "One-Way Hash Functions," *DDJ*, September 1991). Robshaw's algorithm, designed jointly with Burt Kaliski, operates on large blocks—1024 bytes—but the principles can be used to design ciphers with smaller

block sizes. It is also likely to be important, as RSA Data Security provides encryption technology to companies such as Microsoft, Lotus, IBM, Novell, and Apple.

Phil Rogaway presented SEAL, a new stream cipher developed jointly with Don Coppersmith, IBM's top cryptographer. This algorithm will be used by IBM for software encryption of disk files in PC-based network software. The algorithm is based on repeated lookup of large tables of pseudorandom numbers.

Hugo Krawczyk of IBM presented implementation and performance details of the Shrinking Generator, which he also designed with Don Coppersmith and first presented at Crypto '93. (See "The Shrinking Generator," *Advances in Cryptology: CRYPTO '93 Proceedings*, Springer-Verlag, in preparation.) The Shrinking Generator is a simple stream cipher which uses the output of a linear-feedback shift register to decimate the output of another linear-feedback shift register.

Markus Dichtl of Siemens presented a derivative of the Shrinking Generator which combines the output of linear-congruential generators rather than linear-feedback shift registers.

Ross Anderson of Cambridge University presented a modern rotor machine. Rotor machines were all the rage in cryptography before algorithms moved to computers. The German Enigma was a rotor machine, as were several U.S. cryptography machines. They have fallen out of fashion recently, but this algorithm is an attempt to show that they can still be secure in the face of massive computing power. Anderson's proposal is for a rotor machine driven by a linear-feedback shift register; it is simple to describe and to implement, and yet seems to be very secure.

David Wheeler, also of Cambridge, proposed a bulk-encryption algorithm based on experience designing algorithms for secure digital telephones. It is based on iterating functions defined by large lookup tables of random permutations, and can perform ultra-fast encryption of large amounts of data.

My own Blowfish algorithm is a 64-bit block algorithm with a variable-length key. (See "Blowfish: A New Encryption Algorithm," on page 38 of this issue.)

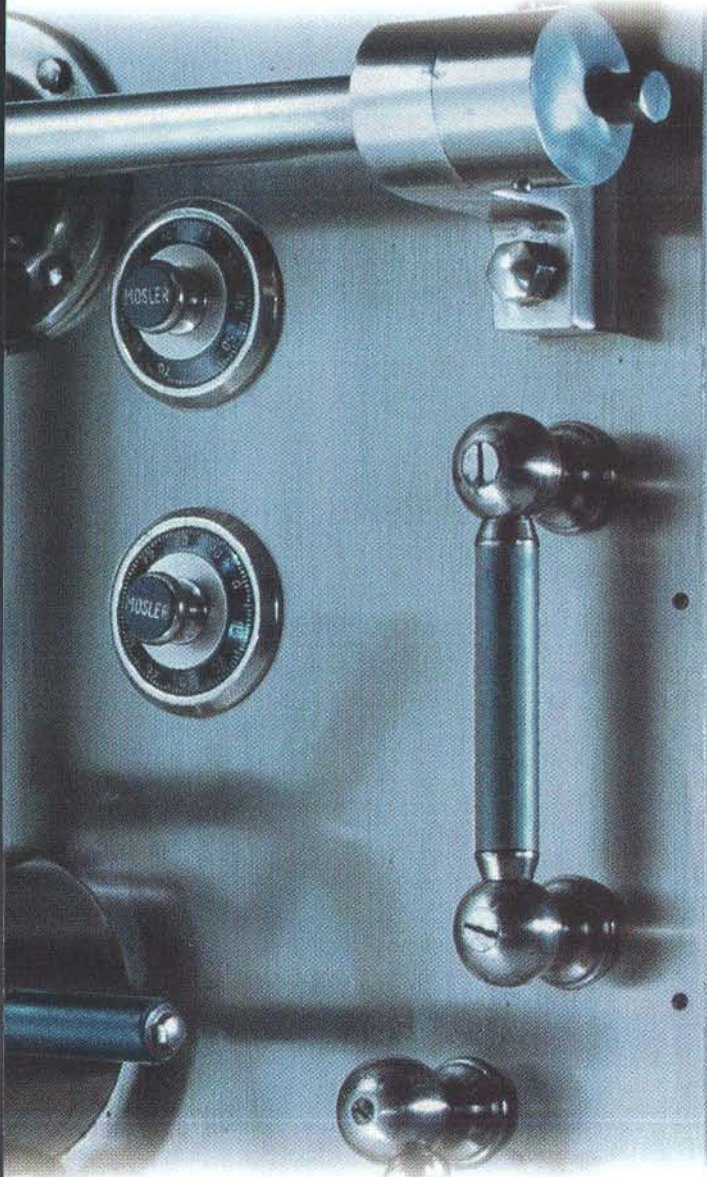
William Wolfowicz (of the Italian telephone company's research lab) presented a 64-bit block algorithm with a 128-bit key which was designed jointly with Adina di Porto. The algorithm makes use of a novel permutation property pointed out by the Russian number theorist Vinogradov.

Joan Daemen presented 3-WAY, a new block cipher he's been developing with colleagues at Leuven, Belgium for two years. It is designed to be fast in both hardware and software, and to resist both differential and linear cryptanalysis attacks.

The fastest algorithm appears to be either Wheeler's or Rogaway-Coppersmith's. Both use about 20 instructions (including four table lookups) to encrypt a 32-bit word: about five clock cycles per byte. On a SparcStation, they had throughput of about 20 Mbytes/second; on a DEC Alpha, about 100 Mbytes/second. However, many of the other algorithms are not far behind.

It is too early to tell if any of these algorithms are secure. The important thing is that they are out there and that cryptanalysts are starting to examine them. I expect at least two of them to fall before the end of the year. (It would be unfair to divulge which I think are insecure.) The techniques used to break them will be recycled into the design of new encryption algorithms, which may then be broken using new techniques. And so the cycle of research will continue.

Hopefully, in five or six years there will be a few algorithms that are still considered secure. These may then be proposed as standards to replace DES and then used to encrypt data far into the next century.



Designing Secure Algorithms

If nothing else, the Cambridge workshop proved that fast, efficient, and safe encryption algorithms are as difficult and challenging to design as ever. The rules of algorithm design are simple. An encryption algorithm should be secure under the following conditions:

- The cryptanalyst (that's the guy trying to break the algorithm) knows all the details of the algorithm. He has some ciphertext, and his job is to deduce the plaintext. (This is called a "ciphertext-only" attack.)
- The cryptanalyst not only has the algorithm and some encrypted ciphertext, but also the unencrypted plaintext. His job is to deduce the key. (This is called a "known-plaintext" attack.)
- The cryptanalyst not only has the algorithm, some ciphertext, and the unencrypted plaintext, but he gets to choose what it is. If there is a particular plaintext sequence that, if encrypted, will easily yield the key, he gets to encrypt that sequence. (This is called a "chosen-plaintext" attack.)

All of these attacks are feasible and have been mounted in the real world. (For historical anecdotes, see *The Codebreakers: The Story of Secret Writing*, by D. Kahn, Macmillan, 1967.) Often, noncryptographers insist that the details of their algorithm should remain secret. From the point of view of security, this is a dubious practice. Security should not depend on the secrecy of the algorithm. If it did, it would be far too vulnerable to a "black bag" attack. A hardware-encryption device can be stolen and reverse-engineered; a software-encryption device can be disassembled. (Even the details of DES were published by the government; see the

National Bureau of Standards, Data Encryption Standard, U.S. Department of Commerce, FIPS PUB 46, January 1977.) Cryptographers have to assume that analysts will have everything but the key, simply because it is prudent to do so.

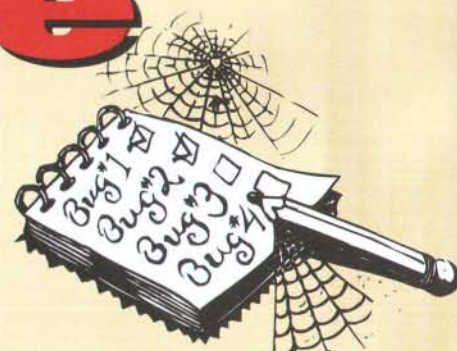
The Skipjack algorithm remains classified and is implemented in the supposedly tamper-resistant Clipper chip. When Intel came up with a new reverse-engineering technique which they thought might beat the tamper protection, the NSA promptly classified it. Even so, the algorithm is probably resistant to anal-

	Key Length	Block Length	Problems
DES	56 bits	64 bits	Key too small
Triple-DES	112 bits	64 bits	Slow
Khufu	64 bits	64 bits	Patented; key too small
FEAL 32	64 bits	64 bits	Patented; key too small
LOKI-91	64 bits	64 bits	Weaknesses; key too small
REDUC II	160 bits	80 bits	Patented
REDUC III	variable	64 bits	Patented
IDEA	128 bits	64 bits	Patented
RC2	variable	64 bits	Proprietary
Skipjack	80 bits	64 bits	Secret algorithm
GOST	256 bits	64 bits	Not completely specified
MMB	128 bits	128 bits	Insecure

Table 1: Block-encryption algorithms. IDEA is used for message encryption in Pretty Good Privacy; PGP. RC2 was developed by RSA Data Security and is used in a variety of commercial software packages. Skipjack is the NSA-developed algorithm in the Clipper chip. GOST is an algorithm developed in the Soviet Union and only recently made public.

BugBase™

has retired the old standard for tracking bugs.



Wave goodbye to pen, paper and home-grown solutions.



BUGBASE™ FOR WINDOWS™ - THE PROFESSIONAL SOFTWARE DEFECT TRACKING TOOL.

- GENERATE POWERFUL REPORTS & GRAPHS
- IMPORT/EXPORT/MERGE DATABASES
- CUSTOMIZE THE SETUP TO YOUR NEEDS
- NETWORK SUPPORT

FOR MORE INFORMATION, CALL:
415-567-4010

IN EUROPE, CALL: + 47 67 15 67 00
FAX: + 47 67 15 67 01

Archimedes Software, Inc. 2159 Union Street, San Francisco, CA 94123

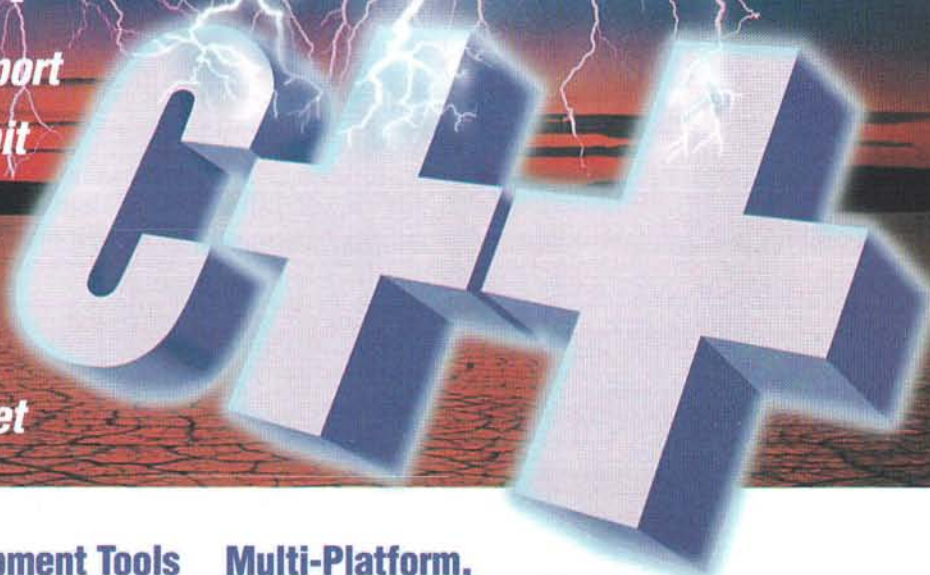
CIRCLE NO. 849 ON READER SERVICE CARD



"At a glance, you can tell the status of any defect or change order, who is responsible for correcting the defect or implementing the change, and when the updated version is expected."

Multi-Platform 32-bit Power: WATCOM C/C++³²

- ▶ **Extensive C and C++ Support**
- ▶ **The Widest Range of 32-bit Intel x86 Platforms**
- ▶ **The Industry's Leading Code Optimizer**
- ▶ **Multi-platform, Cross-development Toolset**



Professional C and C++ Development Tools

C/C++³² delivers the key technologies for professional developers: comprehensive C++ support including templates and exception handling; advanced superscalar optimization; and 32-bit multi-platform support. C/C++³² includes both C and C++ compilers, so you can incrementally adopt the benefits of C++.

Unleash 32-bit Power!

C/C++³² delivers 32-bit performance. The 32-bit flat memory model simplifies memory management and lets applications address beyond the 640K limit. Powerful 32-bit instruction processing delivers a significant speed advantage: typically a minimum 2x processing speedup.

A C++ compiler designed to deliver on the promise of object-oriented programming

The C++ compiler provides comprehensive support for the AT&T version 3.0 language including templates, plus exception handling. These features are key to realizing the benefits of object-oriented programming: code reusability, increased reliability and reduced maintenance.

Hot New Superscalar Code Optimizer

The hot, new C/C++³² code generator advances the performance envelope. New superscalar optimization strategy uses "riscification" and instruction scheduling to deliver improved performance on 486 and Pentium processors. The compiler can create a single, high-performance executable which runs on 386, 486 and Pentium processors.

Industry Standard. Industry's Choice

WATCOM's working relationships with industry leaders such as Autodesk, GO, IBM, Intel, Lotus, Microsoft and Novell ensure that we continue to understand and meet the needs of the software industry and professional developers.

Multi-Platform, Cross Development Support

C/C++³² supports a wide range of 32-bit Intel x86 host and target platforms allowing professional developers to leverage the multi-platform, cross-development capabilities of today's operating environments including OS/2 2.x and Windows NT.

- ▶ **32-bit DOS host and target support;** includes the DOS/4GW 32-bit DOS extender by Rational Systems with royalty-free run-time and virtual memory support up to 32Mb.
- ▶ **OS/2 2.x host and target support;** enables the development of OS/2 2.x applications and DLLs. 32-bit Windows 3.x applications can exploit WIN-OS/2 using WATCOM's Windows Supervisor technology.
- ▶ **Windows NT host and target support;** includes a Windows NT-hosted debugger for 16-bit Windows 3.x, Win32s and Windows NT debugging; also includes Win32s target support.
- ▶ **32-bit Windows 3.x target support;** includes WATCOM's 32-bit Windows Supervisor, and enables development and debugging of true 32-bit Windows 3.x applications and DLLs.

Also Available: WATCOM C³² for DOS

C³² for DOS is a professional, low cost 32-bit C compiler and tools package enabling development, debugging, performance profiling and royalty-free distribution of 32-bit applications for extended DOS. Suggested retail price: \$199*.

WATCOM C/C++³² has a suggested retail price of \$599*.

For additional information or to order direct call 1-800-265-4555. Call our FAX Back system at 1-519-747-2693 from your fax machine for immediate product information.

WATCOM

CIRCLE NO. 698 ON READER SERVICE CARD

1-800-265-4555

The Leader in 32-bit Development Tools

415 Phillip Street, Waterloo, Ontario, Canada, N2L 3X2 Telephone: (519) 886-3700, Fax: (519) 747-4971

*Price in US dollars. Does not include freight and taxes where applicable. Authorized dealers may sell for less. WATCOM C and the Lightning Device are trademarks of WATCOM International Corp. DOS/4G and DOS/16M are trademarks of Rational Systems Inc. Other trademarks are the properties of their respective owners. Copyright 1993 WATCOM International Corp.



(continued from page 20)

ysis even if its details become known. (The primary reason that NSA does not want to release the inner workings of Skipjack is probably because they don't want their secret techniques used to design other high-security algorithms.)

Cryptographers also have to assume that analysts have access to enormous amounts of computing power—more computing time than can be optimistically expected during the next 100 years or so. In the face of all these conditions, the algorithm should still be unbreakable.

The goal of the conference was to propose algorithms capable of encrypting data at a dozen or so clock cycles per byte

Key Length

Key length is a poor measure of the security of an algorithm, but it's a good place to start. Algorithms with long keys are not necessarily secure, but algorithms with short keys are definitely insecure.

For instance, earlier this year Michael Weiner presented a design for a brute-force DES cracker (see "Efficient DES Key Search," *Advances in Cryptology: CRYPTO '93 Proceedings*, Springer-Verlag, in preparation). He didn't just do some theoretical calculations, but went through the entire design process. He designed a custom cracking chip down to the gate level and had his in-house fabrication department estimate fabrication costs. He designed a controller board, had its cost estimated, and designed and priced peripheral hardware (power supplies, racks, and a complete mechanical housing). What Weiner determined was that with a \$1 million machine, he could break DES in three-and-a-half hours. This is cheap enough to hide in the budget of several different government agencies. It's even cheap enough to be considered by large corporations or organized-crime syndicates. (His employer, Bell Northern Research, claims to have no interest in building a working model.)

This work has important implications for algorithm design. There's nothing special about DES; the analysis will be similar for all algorithms. 56 bits is too small for a key; even 64 bits is too small. Even 80 key bits is marginal—only enough for short-term security. Any algorithm proposed today should have a key length of at least 128 bits; see Table 2.

These calculations are based on present-day computers. For future projections, plan on computing power doubling every 18 months. Each of the above numbers becomes an order of magnitude smaller every five years: A \$1 billion machine that takes 6.7 years to break a key today will take 0.67 years (8 months) with 1999 technology and 0.067 years (24 days) with 2004 technology. What is secure now might not be in 50 years. In light of these calculations, Skipjack's 80-bit key seems woefully inadequate.

Key length is also critical for export. The U.S. Government does not permit the export of algorithms with key lengths greater than 40 bits. (Yes, some exportable algorithms appear to have longer key lengths, but the effective key length is 40 bits or less.) Various computer-privacy advocates are trying to change this.

Some of the algorithms presented at the Cambridge workshop had variable-length keys. This is especially desirable because it allows the implementor to define his own level of security. If he has to export the algorithm, the implementor can set the key length at 40 bits. For low-grade security (information that only has to remain secret for a few minutes), you can use 64 bits. If you need long-term security, you can use key lengths of 128 or even 256 bits.

Variable-length keys were generally constructed during a "key-expansion phase" of the algorithm. Generally, there was an initial bit of computation required before the algorithm could encrypt any data. During this computation, the key typed in by the user would be expanded into a large set of subkeys used for encryption. DES does this to some degree; the 56-bit key is expanded into an array of subkeys totaling 768 bits. Some algorithms at the Cambridge workshop took this to an extreme, expanding a key into subkey arrays totaling 1 Kbyte of data or even more.

Differential and Linear Cryptanalysis

Of course, the trick to algorithm design is to make sure that a brute-force attack is the most efficient way of getting the key, although for most encryption algorithms, there are other ways. These methods, generally very complex and mathematical, involve exploiting the structure of the algorithm.

Differential cryptanalysis and linear cryptanalysis are two new attacks that have been successfully used against DES and other algorithms. Differential cryptanalysis was invented by Biham and Shamir in 1991 (see *Differential Cryptanalysis of the Data Encryption Standard*, by E. Biham and A. Shamir, Springer-Verlag, 1993), while linear cryptanalysis was invented by Mitsuru Matsui in 1993 (refer to "Linear Cryptanalysis Method for DES," *Advances in Cryptology: CRYPTO '93 Proceedings*, Springer-Verlag, in preparation). Differential cryptanalysis is a chosen-plaintext attack: It looks at differences between pairs of plaintexts and corresponding pairs of ciphertexts. These differences, along with information about the structure of the underlying algorithm, give an analyst clues about the key. Collect enough of these differences, and you can find the key more efficiently than you would with brute force.

Don't get too excited, though. The best chosen-plaintext differential cryptanalysis attack against DES has a complexity of 2^{47} . This is better than the 2^{56} required for brute force, but requires on the order of 10 terabytes of chosen-plaintext data. Although interesting, it is still more theoretical than practical. The best way to attack DES is still brute force.

Key Length	Time for a \$1M Machine to Break	Time for a \$1B Machine to Break
40 bits	0.2 seconds	0.0002 seconds
56 bits	3.5 hours	13 seconds
64 bits	37 days	54 minutes
80 bits	2000 years	6.7 years
100 bits	7 billion years	7 million years
128 bits	10^{18} years	10^{15} years
192 bits	10^{37} years	10^{34} years
256 bits	10^{56} years	10^{53} years

Table 2: Key length and security in 1994.

Attack	Type	Complexity
Brute-force	Known-plaintext	2^{55}
Differential	Known-plaintext	2^{55}
Differential	Chosen-plaintext	2^{47}
Linear	Known-plaintext	2^{43}

Table 3: Cryptanalysis of DES.

"You asked for a powerful and affordable tool to develop client/server applications. That's why I developed System Architect 3.0."

JAN POPKIN, CHIEF SCIENTIST
POPKIN SOFTWARE & SYSTEMS, INC.

Developers and project teams looking for a CASE-based tool for client/server application development will find the answer in System Architect™ 3.0. This latest version of the CASE price/performance leader includes the features of expensive tools for a fraction of the cost.

System Architect 3.0.

SA 3.0 simplifies the development of client/server applications by supporting multiple methodologies including Information Engineering, Gane & Sarson, Yourdon, IDEF, OOA&D, SSADM IV, Shlaer/Mellor, and Ward & Mellor. It also features an integrated data repository you can customize. And it runs under MS Windows® or IBM's OS/2 PM.®

Flexibility And Functionality.

The ideal combination of flexibility and functionality has made SA the undisputed price/performance leader. As the needs of developers have changed, so has the scope of SA's features and options:

SA Screen Painter: Allows repository-based development of GUI screens and menus or character-based screens.

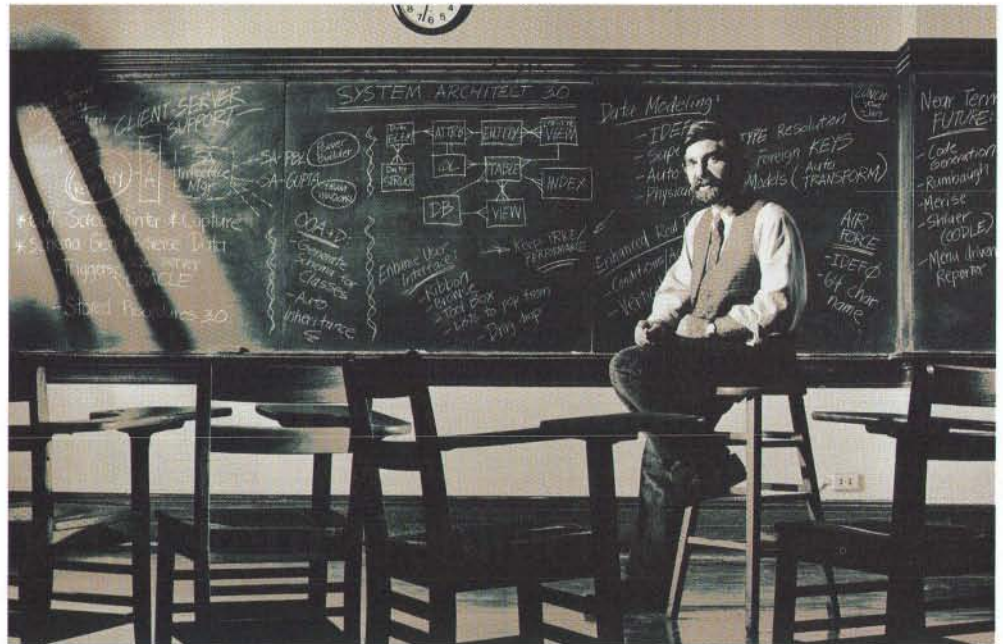
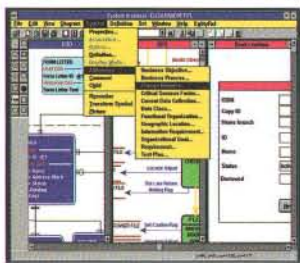
SA Object-Oriented Version: Supports Booch '91 and Coad/Yourdon.

SA Reverse Data Engineer: Reverse engineers SQL databases, including SQL Server, SYBASE, DB2, Informix, and Oracle.

SA Schema Generator: Generates DDL and SQL triggers from entity models for Oracle, Informix, Ingres, PROGRESS, Paradox, dBASE III, DB2, SQL Server, SYBASE, and other SQL and 4GL databases.

SA Project Documentation Facility: Enables the automatic generation of deliverables with desktop publishing quality from SA Encyclopedia.

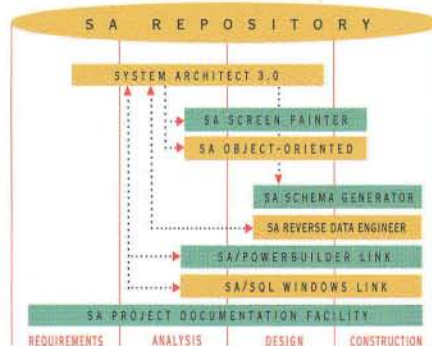
Paint GUI screens from data in repository.



Choose Your Development Environment.

SA/PowerBuilder Link: Allows the exchange of design information between SA and PowerBuilder for the development of more robust client/server applications.

SA/SQL Windows Link: Works with Gupta's SQL Windows.



▲ System Architect 3.0 covers your development lifecycle with a complete range of features and options.

Put Your Project Team In A Class Of Its Own.

System Architect 3.0 makes your project team more productive with a range of capabilities including:

Network Version: Allows multiple team members to work concurrently on a project while

sharing the SA Repository by locking diagram and data dictionary records.

Network Security: Allows Project Managers to uniquely identify and classify personnel with appropriate levels of authorization.

Access Control: Allows team members to check-out, check-in, or freeze encyclopedia objects with defined authorization.

Version Control: Allows project encyclopedias, and their related files, to be saved and stored with appropriate version-identifying data. (Available in version 3.1)

**Call Us Today At
800-REAL-CASE, X131.**

To find out how to qualify for your free 30-day evaluation copy, call us today, or fax us at 212-571-3436.

SYSTEM ARCHITECT 

Popkin Software & Systems, Inc.,
11 Park Place, New York, NY 10007

England 44-926-881186; Benelux 31-3406-65530;
Germany 49-6151-82077; Italy 39-49-8700366;
Switzerland 41-61-6922666; Denmark 45-45-823200;
Australia 61-02-346499; Sweden 46-8-626-8100;
Elsewhere 1-212-571-3434

Read this!

Introducing:

DYNACE™ Dynamic C language Extension

Dynace (pronounced like "dynasty" without the "t") is a new object oriented extension to the C language. Dynace extends C like C++, however, Dynace is easier to learn, has stronger OO facilities, is much better suited to the development and maintenance of large applications and is more portable than C++.

- ☒ No preprocessor or C syntax changes - much easier to learn than C++ for C programmers!
- ☒ Compiles with a normal C compiler - not an interpreted language - runs fast!
- ☒ Multiple inheritance, metaclasses, generic functions, dynamic binding, automatic garbage collection, multiple threads, named pipes, counting semaphores, powerful class library!!
- ☒ Links seamlessly with regular C code - your pre-existing C code may still be used - easy to incrementally add OO facilities to your app!
- ☒ Full, portable C source available - no longer be the victim of a proprietary solution!
- ☒ Prior knowledge of object oriented concepts not necessary - manual fully describes all concepts - many examples included!

Introductory pricing:

Developer edition	\$ 99.00
Source edition	499.00

Dynace has many more features which we don't have room to explain. Please call us to receive more information, free and at no obligation.



Algorithms Corporation
3020 Liberty Hills Drive
Franklin, TN 37064

615-791-1636 Voice (collect)
615-791-7736 Fax

CORPORATION™

© 1993 Algorithms Corporation

CIRCLE NO. 912 ON READER SERVICE CARD

(continued from page 22)

Linear cryptanalysis is similar to differential cryptanalysis, but looks for linear relationships between selected bits of the plaintext, ciphertext, and key. Against DES, this attack has a complexity of 2^{43} . Even better, it is a known-plaintext attack. However, it still requires much too much data to be practical.

Now that these attacks are known, there are techniques for optimizing encryption algorithms so that they are resistant to them. Most of the algorithms presented at the workshop took these attacks into account during the design process; see Table 3.

Cascading Multiple Algorithms

One way to increase the security of your system is to chain multiple algorithms together. For example, first encrypt your file with DES and one key, and then with IDEA (see "The IDEA Encryption Algorithm," by Bruce Schneier, *DDJ*, December 1993) and another key. The result will be much stronger than using either of the two algorithms individually.

Cascading multiple algorithms might also be the best way to negotiate security with algorithms that some people don't trust. If Alice and Bob want to communicate with each other and don't trust each other's algorithms, they can use both—first her algorithm, and then his. This idea, suggested by Whitfield Diffie, was discussed at the Cambridge workshop.

This sounds good, but there is a problem: Massey and Maurer proved that a cascade of multiple algorithms is at least as strong as the first or, with stream ciphers, at least as strong as the best (see "Cascade Ciphers: The Importance of Being First" by Maurer and Massey, *Journal of Cryptology*, 1993). The difficulty with proving anything more than this is that a bad guy might provide you with a first algorithm which twisted your plaintext around so as to provide a chosen-plaintext attack on the second algorithm which you supplied.

This applies not just to encryption algorithms, but to any process designed by someone else which you incorporate into your system. In fact, the widely used CELP code (which compresses digital speech to modem speeds) was designed by the NSA, and for all anyone knows it could be acting as a cryptanalyst's helper in some subtle way. It does seem though, that a cascade of algorithms is better than individual algorithms, provided that the second and subsequent algorithms are secure against chosen-ciphertext attacks, and provided all the algorithms' keys are independent.

The real benefit of cascading algorithms is in design diversity; it makes the overall system less vulnerable to a cryptanalytic attack. Both triple-DES and IDEA seem secure today, but there is always the possibility that some clever mathematician might come up with a good attack against one of them tomorrow. Using triple-DES, then a fast stream cipher such as Wheeler's algorithm, and then IDEA, would be immune to new attacks against any one of the three algorithms. Successful attacks against all three would be required to break the cascaded system.

Conclusion

Encryption algorithms are like airplanes. It's easy to design one, but it's hard to design one that flies. To make matters worse, it's hard to tell if any one of them is any good. The only real way to test the security of an algorithm is to let other programmers try breaking it. But even if the algorithm survives years of intense analysis by many different people, you can still only hope that it is really secure.

DDJ

To vote for your favorite article, circle inquiry no. 1.

One source is all you really need.



The news today is the Wind/U 2.0™ portability toolkit! Wind/U enables your Microsoft Windows and Visual C++ applications to run as native UNIX/Motif applications. You can use your Microsoft Windows source base to rapidly penetrate the fast growing workstation marketplace.



The advantage of Wind/U is that the toolkit uses the complete set of the Microsoft Windows application programming interface (API) facilities under Motif. You only need to recompile your Windows

application in the UNIX environment and link to the Wind/U Library. Wind/U maps the Windows function calls to Motif and X calls, allowing your Windows application to execute as a native Motif application.

Bristol Technology Inc.

Bringing the Best of Microsoft Windows to X and Motif

Wind/U ported applications maintain identical functionality between Windows and Motif versions since Wind/U provides features such as on-line help, PostScript and PCL printing, DDE, MDI, and Common Dialogs. Wind/U supports Win16, Win32s, and the Microsoft Foundation Class Library.

Now you can cost-effectively run your applications on Sun, IBM, Hewlett-Packard, and DEC workstations, while focusing your programming efforts on the popular Windows API.



For more information call (203) 438-6969, email to info@bristol.com or fax to (203) 438-5013.

CIRCLE NO. 851 ON READER SERVICE CARD

Wind/U is a trademark of Bristol Technology Inc. UNIX is a registered trademark of Novell. Microsoft is a registered trademark and Windows and Visual C++ are trademarks of Microsoft Corporation. Motif is a registered trademark of the Open Software Foundation. All other products mentioned herein are trademarks of their respective owners.

Cryptography Without Exponentiation

Secure alternatives to RSA

Peter Smith

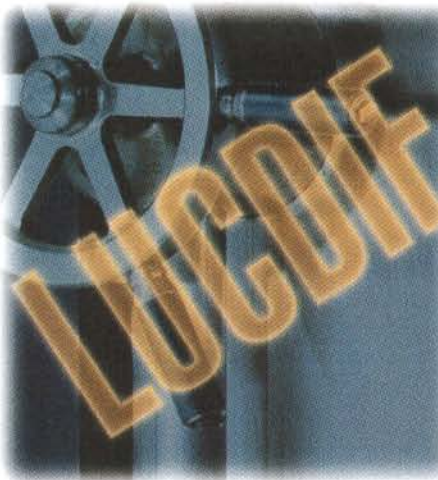
While not a full-fledged public-key cryptosystem, the 1976 Diffie-Hellman key-negotiation technique featured the first cryptographic use of modulus exponentiation. Diffie and Hellman's method, which is used to establish a secret key over an insecure channel, is still in use because the mathematical problem on which it is based remains as difficult today as it was in 1976. In this, Diffie and Hellman were also the first to base cryptosystems on problems that mathematicians have been unable to solve.

In the January 1993 issue of *DDJ*, I presented an alternative to the RSA encryption algorithm called LUC (see "LUC Public-Key Encryption," *DDJ*, January 1993). As that article suggests, many ciphers, including the Hellman-Diffie-Merkle key-exchange system and the El Gamal digital signature, can be reinforced by replacing the process of exponentiation with the process of calculating Lucas functions. This article extends LUC with three new cryptosystems: the Lucas-function El Gamal public-key encryption, the Lucas-function El Gamal digital signature, and a Lucas-function-based key-negotiation method called LUCDIF.

Peter has worked in the computer industry for 16 years, and served as deputy editor of Asian Computer Monthly. He invented LUC in 1991 and founded LUCENT to commercialize Lucas-function-based cryptography. He can be reached at 25 Lawrence Street, Herne Bay, Auckland, New Zealand.

The Algorithms

The exponentiation ciphers here are all based on the mathematical problem known as the Discrete Logarithm (DL). Basically, this problem reduces to solving for x in the equation $a^x = b \bmod c$; where a , b , and c are integers and their values are known. The cipher known as El Gamal and its variants were introduced over the course of the 1980s and are based on the DL problem. One of these, Schnorr's variant of the El Gamal digital signature, was chosen by the National Institute of Standards and



Technology as the basis of the Digital Signature Standard.

As suggested in my previous article, ciphers based on the DL problem can be implemented using Lucas functions instead of exponentiation. Such implementations are sometimes not without their complications in terms of storage and timing overheads, but they can be shown to be asymptotically as fast. More importantly, they are cryptographically stronger than their exponentiation-based ancestors. It is an open question how much stronger the Lucas-function ciphers are. The fastest known subexponential-time algorithms for attacking the DL can't be used against them, making them vulnerable only to exponential-time attacks.

The mathematical problem on which the Lucas-function ciphers are based is analogous to the DL problem, except that here the problem is to solve for x in the equation $V_x(a,1) = b \bmod c$. This problem has the advantage that the subexponential algorithms do not appear to generalize to it, so breaking these ciphers is much more expensive.

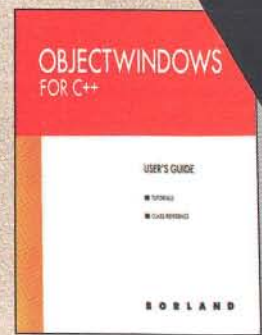
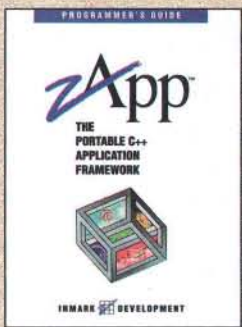
Key Negotiation

The Diffie-Hellman key-negotiation process allows two correspondents, Alice and Bob, to establish a common cryptographic key between them, even if an eavesdropper is listening in on their connection. They both agree on a prime p and a primitive root (or generator) α . Using a secret number A , Alice publishes her part of the key, as given by $\alpha^A \bmod p$. Similarly, Bob publishes his part of the key using his secret number B , using the formula $\alpha^B \bmod p$. In Alice's case, she takes Bob's key and forms $(\alpha^B)^A \bmod p$, while Bob takes Alice's published key, and forms $(\alpha^A)^B \bmod p$. Since $(\alpha^B)^A \bmod p$ equals $\alpha^{AB} \bmod p$ equals $(\alpha^A)^B \bmod p$ equals some value K , say, then both Alice and Bob now have the same key. This method, the first successful, though partial, implementation of public-key ideas, lets part of the key be made public.

The DL problem seems to guarantee that an eavesdropper, who has only public knowledge and not the secret values A and B , cannot find K . If p , A , and B are large enough (say, over 500 bits in length), there is only a small chance of guessing the secret values. If the key were to be used as a DES key, Alice and Bob could agree to take only the first 56 bits to K .

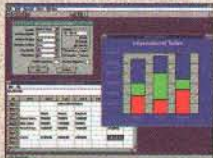
We have called our Lucas-function-based key-negotiation method LUCDIF, combining LUCas and DIFfie. As with LUC, the known multiplicative attacks on Diffie-Hellman do not carry over to LUCDIF, since it is not multiplicative.

LUCDIF is quite analogous to Diffie-Hellman. Choose the prime p in the same way. A value λ must be chosen so

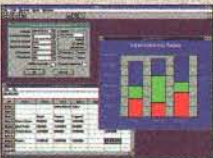


VERY PORTABLE

BARELY PALATABLE



WINDOWS



WINDOWS NT



OS/2 2.x



UNIX X/MOTIF



DOS GRAPHICS



DOS TEXT

Ever get the feeling that OWL stands for "Outdated Windows Library"? That MFC really means "Microsoft Frustration Classes"? If so, toss them out and join the thousands of programmers who have discovered a better application framework: **zApp**.

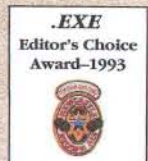
In an industry that's always changing, the future belongs to programs that can change with it. That's why we've developed **zApp**, the industry leading C++ GUI class library. **zApp** allows you to support all the platforms you need with just one set of source code. You simply make calls to **zApp** classes, without worrying about the platform's specific API, and **zApp** does the rest.

zApp is also the most full-featured application framework available. Giving you more classes than OWL and MFC combined, **zApp** sports an unmatched list of capabilities for creating first-class applications quickly and easily. That's one of the reasons why *Windows Tech Journal* wrote "zApp is simply the best designed application framework on the market. If C++ is your language of choice, then you should be using **zApp**. Period".

Finally, **zApp** makes programming easier. By providing an intuitive class structure and plenty of example programs, **zApp** allows you to produce more code faster, making it ideal for C++ programmers of all levels. With over 1200 pages of documentation, including extensive tutorials and demos, as well as free technical support, you know you'll never be left hanging.

Unix X/Motif now available for:
HP HPUX 9.x – IBM RS/6000 AIX – SCO
SunOS 4.1.x – Solaris 2.x – Solaris x86

So pick up the phone now
 and call 800-346-6275
 and get a glimpse of what
 the future has to offer.



- Source Code Included
- No Runtime Royalties
- Compatibility with Most Major Compilers
- Documentation - including over 65 sample programs, tutorials and 880 page Programmers Reference
- Over 35 Window Classes - including application, Dialog, MDI, all Standard Controls, Common Dialogs, and access to custom controls

- Data Entry Classes - includes Text, Picture String, Numeric, Date, Time, Radio Group, Checkbox and List Fields
- Graphics classes - including Windows, Bitmap, Printer, Fonts, Pens, Brushes, Bitmaps, Colors, and Regions, Printer and Font enumeration
- Object Persistence - Using a streams interface, easily store objects and other data to disk and retrieve them
- Advanced MDI Support
- Robust and Efficient Memory Management

- Advanced Message Handling
- Advanced Debugging Support - traps many common programming errors - includes a powerful assert system
- Dynamic Sizers - Sizers provide automated window and graphical object layout management
- Advanced Scrolling Classes
- Menus - including standard, system, and popup
- Full DDE Support
- Much, much more

2065 Landings Drive, Mountain View, CA 94043

Tel: 800-346-6275 or 415-691-9000 • Fax: 415-691-9099 • BBS: 415-691-9990

In the U.K., call Systemstar at (0992) 500919 • In Germany, call ESM Software at 07022-9256-0

In Italy, call Sartocomputer at (049) 654221 • CompuServe: GO INMARK • Internet: sales@inmark.com



from page 26)

dition in Figure 1(a) is true. With a value is easy. Every value has a 50 percent chance of satisfying the condition. Now the values given by $V_A(\lambda, 1) \bmod p$ and $V_B(\lambda, 1) \bmod p$ are published by Alice and Bob,

(a) $V_{(p+1)/2}(\lambda, 1) \neq 2 \bmod p$, for all $t > 1$ dividing $(p+1)$

(b) $V_{nm}(P, Q) = V_n(V_m(P, Q), Q^m)$ Relation 1
 $2V_{n+m} = V_n V_m + DU_n U_m$ Relation 2

Figure 1: (a) Choosing a value λ ; (b) Lucas-function relations which let us transform exponentiation to Lucas-function calculation.

respectively. Bob takes Alice's number and calculates $V_B(V_A(\lambda, 1)) \bmod p$. Similarly, Alice calculates $V_A(V_B(\lambda, 1)) \bmod p$.

Relation 1 in Figure 1(b) shows that these two values are the same and that Alice and Bob have obtained the same key, K' . If p is a prime of over 512 bits, then this method of key negotiation is very secure. Once again, for a DES key, Alice and Bob may decide to select only the first 56 bits of K' .

El Gamal and LUCELG

The El Gamal cipher comes in two parts. There is a procedure for encrypting and decrypting and a second procedure for signing and verifying a digital signature.

For encryption, assume Alice wants to send a message M to Bob using his public key y which is equal to $\alpha^x \bmod p$ (x is Bob's private key). Alice first finds a secret number k , which is greater than zero and less than p , and calculates L using $L \equiv y^k \bmod p$. Two other values are then worked out: $c_1 \equiv \alpha^k \bmod p$, and $c_2 \equiv LM \bmod p$. These two values, c_1 and c_2 , make up the cryptogram which Alice sends to Bob.

Schnorr's variant of the El Gamal digital signature was chosen as the basis of the Digital Signature Standard

For decryption, Bob first calculates L using the fact that $L \equiv (\alpha^k)^x = (c_1)^x$, since only he knows the value of his secret key x . Having found L , Bob calculates its multiplicative inverse (L^{-1}), and multiplies this by c_2 , recovering M ; $M \equiv c_2(L^{-1}) \bmod p$.

The Lucas-function version of El Gamal public-key encryption and decryption follows a path similar to that of El Gamal public-key encryption/decryption. Bob's public key, in this case, is $V_x(y, 1) \bmod p$. A secret value k is also necessary here, and we first calculate G . When encrypting, $G \equiv V_k(y, 1) \bmod p$. The two halves of the cryptogram are then computed: $d_1 \equiv V_k(y, 1) \bmod p$, and $d_2 \equiv GM \bmod p$.

In the decryption case, Bob deciphers the cryptogram by solving for G ; $G \equiv V_x(d_1, 1) \bmod p$. The multiplicative inverse of G can be calculated, modulo p , using the extended Euclidean algorithm (see Knuth), and the message is recovered by $M \equiv d_2(G^{-1}) \bmod p$. Figure 2 provides an example.

Note that the LUCELG cryptogram is twice the size it would be in LUC. Both d_1 and d_2 almost always have the same number of digits as the modulus, so the combined cryptogram will have a length of about twice that of p . This is also the case with the exponentiation version.

Digital Signature

The El Gamal digital signature is more cumbersome to convert from exponentiation to Lucas functions than is El Gamal public-key encryption/decryption. However, observing that Lucas

COMPUTER LANGUAGE PRODUCT EXCELLENCE AWARD 1991 PC-lint for C/C++ presents Bug # 785

```
#include <stdio.h>

class complex
{
public:
    double real, imag;
    operator double() { return real + imag; }
};

complex x;

int main()
{
    x.real = 1.0; x.imag = 1.0;
    complex cc[2] = { x, x };
    printf( "%g %g \n", cc[0].real, cc[1].real );
    return 0;
}
```

Instead of printing "1 1" as expected, this program prints "2 0". Why? Call if you need a hint. Refer to Bug #785.

PC-lint for C/C++ will catch this and many other bugs. It will analyze a mixed suite of C and C++ modules to uncover bugs, glitches, quirks and inconsistencies.

Numerous C++ Warnings and Messages:

Are your inherited destructors virtual? Are your constructor **new**'s matched by your destructor **delete**'s? Are your initializers in order? Are names inadvertently hiding other names? Are your C++ modules consistent with your C modules? Much, much, more.

Plus Our Traditional C Warnings:

Uninitialized variables, unaccessed variables, possibly uninitialized variables, strong type mismatches, indentation irregularities, loss of precision, strange uses of Booleans, signed/unsigned mismatches, suspicious expressions, unused macros, etc. etc.

Full C++ Support - PC-lint for C/C++ is based on the ARM and is tracking the latest ANSI/ISO draft including exceptions and templates. It supports both Borland and Microsoft C/C++.

Options Galore: A plethora of options for message suppression, message format, compiler dependencies, etc. All messages can be individually suppressed or enabled, both locally and globally. Numerous compilers/libraries supported. Runs on MS-DOS (Optional built-in 386 DOS extender) and OS/2.

PC-lint for C \$139
PC-lint for C/C++ \$239

PC-lint users: call for update pricing
Unix and Mainframe programmers: call for pricing for FlexLint.

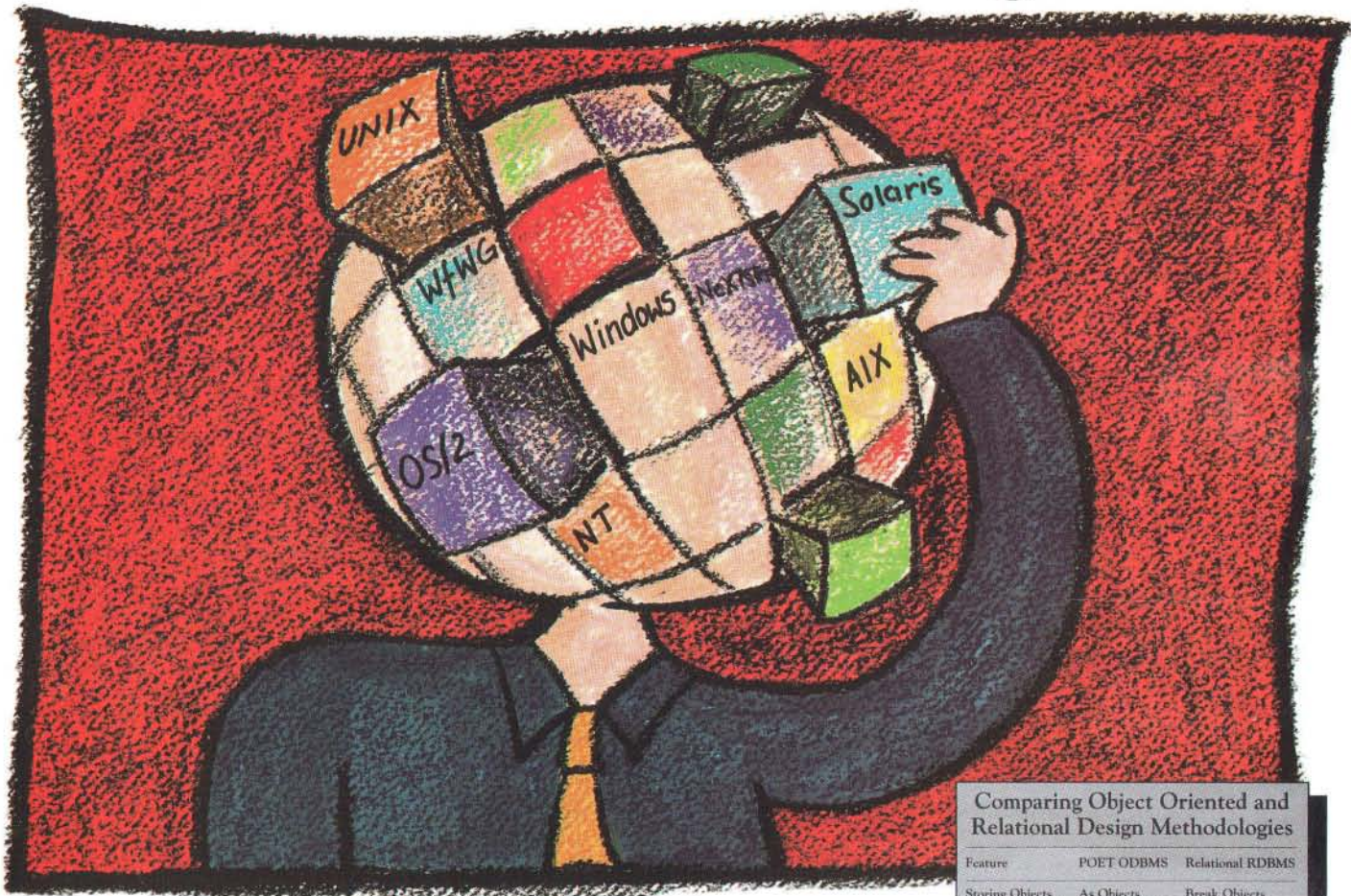
Gimpel Software

3207 Hogarth Lane, Collegeville, PA 19426
CALL TODAY (610) 584-4261 Or FAX (610) 584-4266
 30 Day Money-back Guarantee.

PA add 6% sales tax.

PC-lint and FlexLint are trademarks of Gimpel Software

With our Object Database, All the Pieces Come Together.



Plus
\$299
Introductory
Offer

Smoothly. Easily. That's the beauty of POET, the Object-Oriented Database System for C++. It's simply a more efficient way to work, and yields finished applications that outperform relational applications in both productivity and performance.

What's the secret of Object Oriented Database programming? The POET ODBMS works with a one-step approach, seamlessly integrating into your application by storing your C++ objects in the database. By contrast, the relational approach forces you to design and maintain separate application and database models and then write lots of code to tie them together. So POET is more direct. More logical. And far more productive.

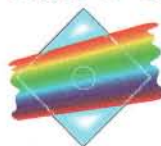
Compare POET with the relational design methodology you're using now.

For a limited time, you can get the POET Personal Edition 2.1 for Windows for just \$299*. Or, call toll free to request a revealing white paper that compares the two methodologies and includes testimonials from key POET installed clients. You'll find that POET is the solution to your application design puzzles.

Comparing Object Oriented and Relational Design Methodologies

Feature	POET ODBMS	Relational RDBMS
Storing Objects	As Objects	Break Objects into Tables
Database Model	User Application Model	Separate Database Model Required
C++ Integration	Total	Poor
Database Operations	At Object Level	Must Write Code
Productivity	Increased	Reduced
Complex Object Performance	Excellent	Poor

Call 1-800-950-8845



POET
Software

*This introductory offer is good through March 31, 1994. †POET applications are portable, at source and database level across all these popular platforms: Windows, OS/2, Macintosh, Windows NT, UNIX (Sun, Silicon Graphics, IBM), NEXTSTEP 486 and Novell (NLM)
POET Software Co. 4633 Old Ironsides Dr., Suite 110, Santa Clara, CA 95054 TEL: (408) 970-4640 FAX: (408) 970-4630

CIRCLE NO. 877 ON READER SERVICE CARD

om page 28)
 e formulas for multiplying
 , subscripts — see Figure
 can construct an El Gamal-
 , since El Gamal's manipula-
 tion of exponents can be converted to

the manipulation of Lucas-function sub-
 scripts. The formula for the addition of
 subscripts (Relation 2) involves the Lu-
 cas $\{U_i\}$ "sister" series. Subsequently, our
 Lucas-function alternative to El Gamal
 involves the doubling of the public-key

Let the prime p be 908797. Choose $k=1949$, $\gamma=19$, $x=2089$, $y=894501$ and $M=1111$.

$G \equiv V_k(y, 1) \bmod p = V_{1949}(894501, 1) \bmod 908797 = 788038$.

$d_1 \equiv V_k(y, 1) \bmod p = V_{1949}(19, 1) \bmod 908797 = 307718$.

$d_2 \equiv GM \bmod p = 788038 \cdot 1111 \bmod 908797 = 338707$.

The cryptogram is the pair $(d_1, d_2) = (307718, 338707)$.

The receiver, who knows that the secret key is 2089, first calculates:

$G \equiv V_k(d_1, 1) \bmod p = V_{2089}(307718, 1) = 788038$

The inverse of this is 518288.

$M \equiv d_2(G^{-1}) = 338707 \cdot 518288 = 1111$

The original message.

Figure 2: Example of LUCELG.

AccuSoft Image Format Library 4.0

Import • Export • Scanning • Conversion • Compression
 Display • Printing • Image-Processing • Special-Effects

Fastest libraries on the market
Guaranteed to read all images in existence

WIN ☐ DOS ☐ VBX ☐ WIN-NT ☐ OS/2 ☐ UNIX ☐ MAC
 WIN Pro Gold ☐ DOS-32 ☐ VBX-32 ☐ and others

When your application demands high quality imaging and top performance, only AccuSoft can deliver. We are the leader in high-performance imaging solutions around the globe. AccuSoft provides the highest quality toolkits with a guarantee that can't be touched!

TIFF

PCX

JPG

TGA

BMP

WMF

- 12 formats supported with auto-detect
- Fastest Group 3, Group 4 and JPEG available!
- Non-standard G3 and G4 files also supported.
- Scanning supported including TWAIN.
- High-speed image display and printing.
- Complete image processing including:
 Rotate, invert, resize, sharpen, blur,
 Roberts Cross, Matrix convolutions,
 Color reduction, edge detection, etc.
- Incredibly easy high-level interface.
- Low-level interface; read & write 1 line at a time.
- All compression algorithms supported.
- Custom versions available.
- 30 day money-back guarantee for 16 bit products.

Call now to order or for more details by fax.

(800) 525-3577

(508) 898-9662 (FAX)

AccuSoft

High Performance Imaging!

AccuSoft Corp. 112 Turnpike Rd. Westborough, MA 01581 (508) 898-2770

GIF

EPS

WPG

DIB

PICT

DCX

size (two Lucas function values, U and V , must be given), as well as increasing the size of the signature, because two "r" (U and V) values are necessary.

The variant of El Gamal chosen as the Digital Signature Standard can be converted in a similar manner. In both cases, we produce ciphers apparently based on a problem for which there is no known subexponential-time attack; hence, they are stronger than their prototypes.

The calculation of the n th Lucas function can be done in $O(\log n)$ operations, which is the same order as the computation of similar exponentials. Heuristics to speed up modular exponentiation can be brought over to the calculation of Lucas functions, if in more complicated form (witness the formula for adding subscripts). These new ciphers can be assured of having performance characteristics similar to those of their progenitors.

Conclusion

For the same level of security, these Lucas-function-based ciphers can be used with a shorter modulus than the exponentiation ciphers. For a 512-bit modulus, the reduction is about one fifth, down to 420 bits, for equivalent cryptographic strength. This reduction increases in size as the modulus grows longer. That only exponential-time attacks are possible on the Lucas-function version of the DL problem ensures attempts to solve it are increasingly more expensive than the subexponential-time attacks possible on the DL itself. We have applied for patents on these algorithms.

Finally, LUC Encryption Technology Ltd. (LUCENT), has been incorporated to license and support cryptographic systems based on Lucas functions. For more information, contact Horace R. Moore, 101 E. Bonita, Sierra Madre, CA 91024.

References

Carey, M.R. and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA: W.H. Freeman, 1979.

Diffie, W. and M.E. Hellman. "New Directions in Cryptography." *IEEE Transactions on Information Theory* (November 1976).

El Gamal, T. "A Public-key Cryptosystem and a Signature Scheme Based on Discrete Logarithms." *IEEE Transactions on Information Theory* (July 1985).

Knuth, D.E. *The Art of Computer Programming: Volume II: Semi-Numerical Algorithms*, 2nd ed. Reading, MA: Addison-Wesley, 1981.

Smith, Peter. "LUC Public-Key Encryption." *Dr. Dobbs Journal* (January 1993).

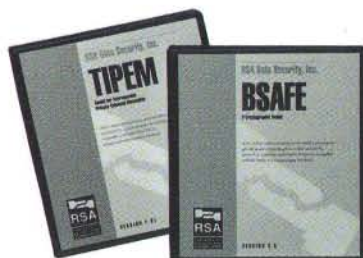
DDJ

To vote for your favorite article, circle inquiry no. 2.



Security Clearance:

The distance between you and the floor after you learn how easy it is to use RSA.



\$290

the industry's leading cryptography shop standing behind your development effort—just like we stand behind other BSAFE™ and TIPEM™ customers like Apple, Microsoft, Lotus and Sun.

Call RSA at 415.595.8782. It could mean the difference between kicking up your heels or just kicking yourself.

RSA. Because some things are better left unread.



Object code versions of RSA toolkits are available for a variety of popular platforms, and source code licensing is also available. Contact RSA for details.
©1993, RSA Data Security, Inc., 100 Marine Parkway, Suite 500, Redwood City, CA 94065-1031

CIRCLE NO. 814 ON READER SERVICE CARD

SHA: The Secure Hash Algorithm

Putting message digests to work

William Stallings

An essential element of most authentication and digital-signature schemes is a hash algorithm. A hash function accepts a variable-size message M as input and produces a fixed-size tag $H(M)$, sometimes called a "message digest," as output (see "One-Way Hash Functions," by Bruce Schneier, *DDJ*, September 1991). Typically, a hash code is generated for a message, encrypted, and sent with the message. The receiver computes a new hash code for the incoming message, decrypts the hash code that accompanies the message, and compares them. If the message has been altered in transit, there will be a mismatch.

The Secure Hash Algorithm (SHA) was developed by the National Institute of Standards and Technology (NIST) and published as a federal information-processing standard (FIPS PUB 180) in 1993. SHA is based on the MD4 algorithm, developed by Ron Rivest of MIT, and its design closely models MD4. SHA is used as part of the new Digital Signature Standard from NIST, but it can be used in any security application that requires a hash code.

William is an independent consultant and president of Comp-Comm Consulting of Breuster, MA. This article is based on material in his forthcoming book, Network and Internetwork Security (Macmillan, due June 1994). He can be reached at stallings@acm.org.

SHA Logic

SHA takes as input a message with a maximum length of less than 2^{64} bits and produces as output a 160-bit message digest. The input is processed in 512-bit blocks. Figure 1 shows the overall processing of a message to produce a digest. The processing consists of the following steps:

Step 1: Append padding bits. The message is padded so that its length is congruent to 448 modulo 512. Padding



is always added, even if the message is already of the desired length. Thus, the number of padding bits is in the range of 1 to 512. The padding consists of a single 1 bit followed by the necessary number of 0 bits.

Step 2: Append length. A block of 64 bits is appended to the message. This block is treated as an unsigned 64-bit integer and contains the length of the original message (before the padding).

The outcome of the first two steps yields a message that is an integer multiple of 512 bits in length. The figure represents the expanded message as the sequence of 512-bit blocks Y_0, Y_1, \dots, Y_{L-1} , so that the total length of the expanded message is $L \times 512$ bits. Equivalently,

the result is a multiple of 16 32-bit words. Let $M[0 \dots N-1]$ denote the words of the resulting message, with N being an integer multiple of 16. Thus $N = L \times 16$.

Step 3: Initialize MD buffer. A 160-bit buffer is used to hold intermediate and final results of the hash function. The buffer can be represented as five 32-bit registers (A, B, C, D, E). These registers are initialized to the following hexadecimal values (high-order octets first):

A=67452301
B=EFCDA89
C=98BADCFE
D=10325476
E=C3D2E1F0

Step 4: Process message in 512-bit (16-word) blocks. The heart of the algorithm is a module that consists of 80 steps of processing; this module is labeled H_{SHA} in Figure 1, and its logic is illustrated in Figure 2. The 80 steps have a similar structure.

Note that each round takes as input the current 512-bit block being processed (Y_q) and the 160-bit buffer value ABCDE, and updates the contents of the buffer. Each round also makes use of an additive constant K_t . Only four distinct constants are used. The values, in hexadecimal, are shown in Figure 3.

Overall, for block Y_q , the algorithm takes Y_q and an intermediate digest value MD_q as inputs. MD_q is placed into buffer ABCDE. The output of the 80th step is added to MD_q to produce MD_{q+1} . The addition is done independently for each of the five words in the buffer with each of the corresponding words in MD_q , using addition modulo 2^{32} .

Step 5: Output. After all L 512-bit blocks have been processed, the output from the L th stage is the 160-bit message digest.

In the logic of each round, each round is of the form:

Accelerate into Client/Server with Paradox

If you're a Paradox® developer, you can easily create client/server applications from your existing code with the new Paradox 4.5 Development Edition for Windows. Now your applications can quickly gain server reliability, security, performance, and expandability.

Paradox: Turn your applications into client/server solutions

Paradox Development Edition gives you all the power and security of Paradox, plus state-of-the-art object-oriented tools—reusable objects that let you quickly create and upsize applications.

You also get extensible tools for reengineering. So you can respond with new solutions to today's fast-changing technology and business trends. And with transparent Paradox-SQL connectivity, you can access data on SQL database servers just as if it were in local PC tables.

InterBase: The most advanced RDBMS for upscaling

Paradox connects to Borland InterBase™ 3.3, which gives you advanced features that put you ahead of the competition. Paradox also connects easily to other popular database servers.



InterBase installs easily with a single command and tunes itself dynamically to the amount of disk and shared memory. Once it's installed, you can immediately use the Paradox SQL Tools included in your Development Edition to create and manage database tables with familiar Windows tools. It's an unbeatable combination.

Borland: Training and support that get you up to speed

Whether you're in the evaluation stage—or building a full solution today—Borland makes it fast and easy to get on-the-spot technical support, with a full range of service options and annual contracts tailored to your special needs.

And with Borland's *automatic* software and documentation maintenance,* you're ensured product updates and upgrades for one year, the minute they're ready! Get Paradox today. It's the fastest way to client/server.



InterBase includes a powerful versioning engine to make your PC applications incredibly fast, even in a mixed read/write environment.

90-day, money-back guarantee!
**See your dealer or call now,
 1-800-458-7730, ext. 8020**
 In Canada call, 1-800-461-3327

Borland
 Power made easy™

(continued from page 32)

$$A, B, C, D, E \leftarrow [CLS_5(A) + f_t(B, C, D) + E + W_t + K_t], A, CLS_{30}(B), C, D$$

where A, B, C, D, E = the five words of the buffer; t = round, or step, number; $0 \leq t \leq 79$; f_t = a primitive logical function; CLS_s = circular left shift (rotation) of the 32-bit argument by s bits; W_t = a 32-bit word derived from the current 512-bit input block; and K_t = an additive constant. Four distinct values are used, and $+$ = addition modulo 2^{32} .

Each primitive function takes three 32-bit words as input and produces a 32-bit word output. Each function performs a set of bitwise-logical operations; that is, the n th bit of the output is a function of the n th bit of the three inputs. The functions are in Table 1. As you can see, only three different functions are used. For $0 \leq t \leq 19$, the function is the conditional function: *If B then C else D*. For $20 \leq t \leq 39$ and $60 \leq t \leq 79$, the function produces a parity bit. For $40 \leq t \leq 59$, the function is True if two or three of the arguments are True.

It remains to indicate how the 32-bit word values, W_t , are derived from the 512-bit message. The first 16 values of W_t are taken directly from the 16 words of the current block. The remaining values are defined as:

$$W_t = W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3}$$

Thus, in the first 16 rounds of processing, the input from the message block consists of a single 32-bit word from that block. For the remaining 64

rounds, the input consists of the XOR of a number of the words from the message block.

SHA can be summarized as:

$$\begin{aligned} MD_0 &= IV \\ MD_{q+1} &= \text{SUM}_{32}(MD_q, ABCDE_q) \\ MD &= MD_{L-1} \end{aligned}$$

where IV = initial value of the ABCDE buffer, defined in Step 3; $ABCDE_q$ = the output of the last round of processing

*SHA is used as part
of the new Digital
Signature Standard
from NIST but can
be used in any
security application
that requires a
hash code*

of the q th message block; L = the number of blocks in the message (including padding and length fields); SUM_{32} = addition modulo 2^{32} performed separately on each word of the pair of inputs; and MD = final message-digest value.

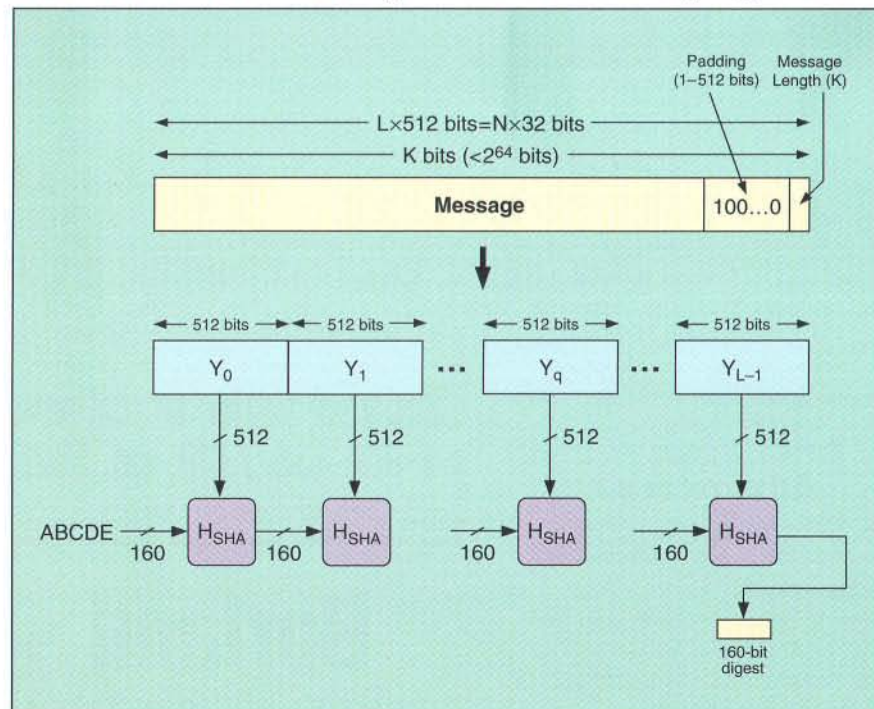


Figure 1: Overall processing of a message to produce a digest.

SHA Security

SHA has the property that every bit of the hash code is a function of every bit in the input. The complex repetition of the basic function f_t produces well-mixed results; that is, it is unlikely that two messages chosen at random, even if they exhibit similar regularities, will have the same hash code. Unless there is some hidden weakness in SHA, which has not so far been published, the difficulty of coming up with two messages having the same message digest is on the order of 2^{80} operations, while the difficulty of finding a message with a given digest is on the order of 2^{160} operations.

DDJ

To vote for your favorite article, circle inquiry no. 3.

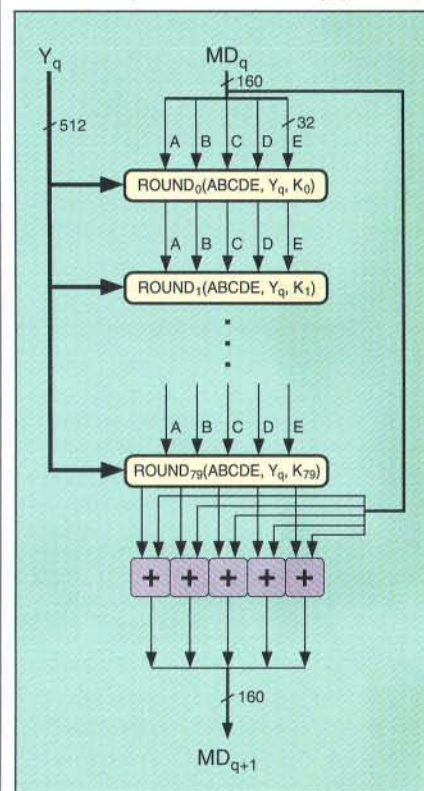


Figure 2: The logic of the module H_{SHA} ; addition (+) is mod 2^{32} .

$$\begin{aligned} 0 \leq t \leq 19 & \quad K_t = 5A827999 \\ 20 \leq t \leq 39 & \quad K_t = 6ED9EBA1 \\ 40 \leq t \leq 59 & \quad K_t = 8F1BBCDC \\ 60 \leq t \leq 79 & \quad K_t = CA62C1D6 \end{aligned}$$

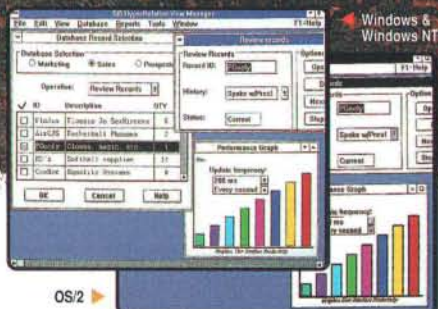
Figure 3: Hexadecimal values of the four constants.

Round	$f_t(B, C, D)$
$(0 \leq t \leq 19)$	$(B \cdot C) \vee (\bar{B} \cdot D)$
$(20 \leq t \leq 39)$	$B \oplus C \oplus D$
$(40 \leq t \leq 59)$	$(B \cdot C) \vee (B \cdot D) \vee (C \cdot D)$
$(60 \leq t \leq 79)$	$B \oplus C \oplus D$

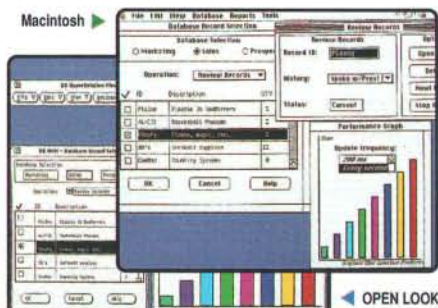
Table 1: SHA primitive functions.

WINDOWS
WINDOWS NT
OS/2
MACINTOSH
OPEN LOOK
OSF/MOTIF
CHARACTER SYSTEMS

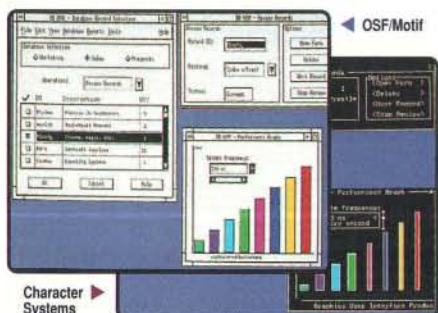
Develop Cross-Platform GUI Applications in a Single Stroke.



OS/2



Macintosh



OPEN LOOK

OSF/Motif

Character Systems

Master the art of multi-platform GUIs.

XVT Software is the leading choice of world-class developers for one reason: It is the simplest, quickest path to building quality applications that port to every GUI without compromises in look-and-feel or performance. Plus, it's easier to learn and use than native toolkits, so your time and effort goes into your application, not your GUIs.

XVT gives you simultaneous original GUIs.

Because XVT uses native GUI objects, your application is indistinguishable from one written directly to the native toolkit. Through our layered architecture, you achieve equivalent cross-platform functionality appropriate to each GUI, without the overhead and inflexibility of proprietary emulation-based systems.

XVT puts complete C/C++ solutions at your fingertips.

XVT Development Solutions for C include an Interactive Design Tool. Solutions for C++ include an object-oriented application framework. Both include the XVT Portability Toolkit.

When combined with in-depth consulting, training and support, plus a wide range of Partners products, XVT forms the most comprehensive and advanced solution for developing completely portable GUI applications.

Developers judge XVT to be a masterpiece.

XVT is the base document for the IEEE's GUI standardization effort. Our thousands of customers include internal and commercial developers like: Alcoa, Amoco, AT&T, Avis, Ford, General Motors, Grammatik/Reference Software, Kodak, Lockheed, NCR, NEC, NIST, Novell, Rockwell, Siemens, Sony, Southwestern Bell, Tandem, Uniplex, Unisys, US Army and US West.

Call now for a Free XVT Demo and Technical Overview.

1-800-678-7988

NEW!
XVT-PowerObjects



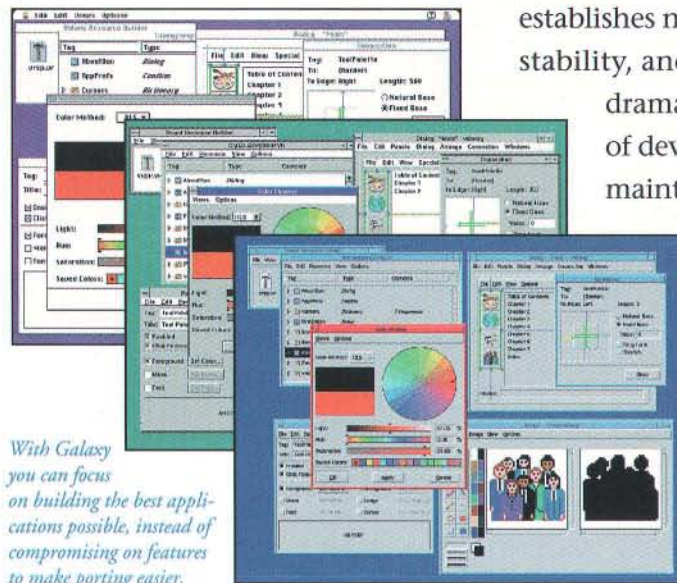
The portable GUI development solution.

XVT Software Inc. 4900 Pearl East Cir. Boulder, CO 80301
(303) 443-4223 FAX (303) 443-0969

For European inquiries, contact: Precision Software GmbH
Phone: 49 0 61 03/37 94 0 Fax: 49 0 61 03/36 95 5

CIRCLE NO. 655 ON READER SERVICE CARD

If you're building a new generation of mission-critical applications, you need a tool with the power and scalability to span departments, divisions—the entire enterprise. Only one tool can help you do it all, because only one tool has it all—the Galaxy Application Environment® from Visix.



With Galaxy you can focus on building the best applications possible, instead of compromising on features to make porting easier.

establishes new standards for power, stability, and completeness—it dramatically reduces the cost of developing, enhancing, and maintaining high-performance distributed applications. Your *entire* Galaxy application can easily be moved to UNIX, Macintosh, Windows, Windows NT, OS/2, and OpenVMS without changing a single line of code.

IT'S ALL IN THE

A New Architectural Standard

If you want to build it once and build it right, then choose your development tool based on what's built-in from the start, not what's bolted-on later. Galaxy is the only application development environment designed specifically for building the next generation of large-scale corporate applications.

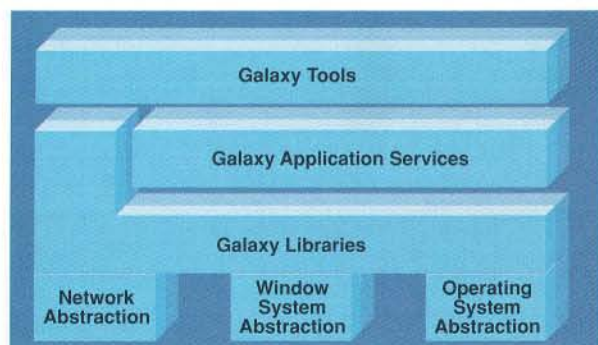
A Complete Solution

In a marketplace crowded with partial solutions such as GUI builders and class libraries, Galaxy stands apart from the pack by delivering a complete multiplatform development environment. Galaxy specifically targets the needs of high-end professional developers by assembling a comprehensive set of innovative tools, libraries, and runtime services into one cohesive development

"An unbelievable product!"
—**UNIX Review**

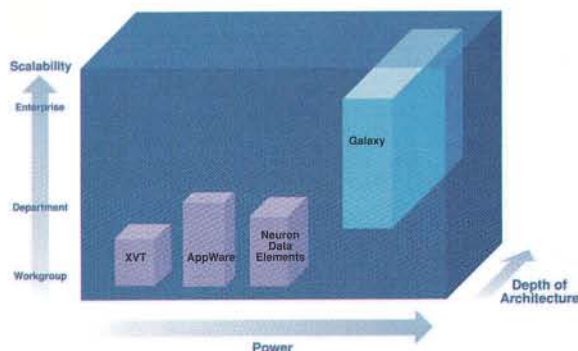


The Galaxy architecture frees you to concentrate on building portable best-of-breed applications, instead of mastering endless operating system, window system, and networking details for each platform. The fully object-oriented Galaxy API



Galaxy's object-oriented architecture offers unprecedented API depth and breadth to liberate C and C++ developers from the constraints of platform-specific toolkits.

visix



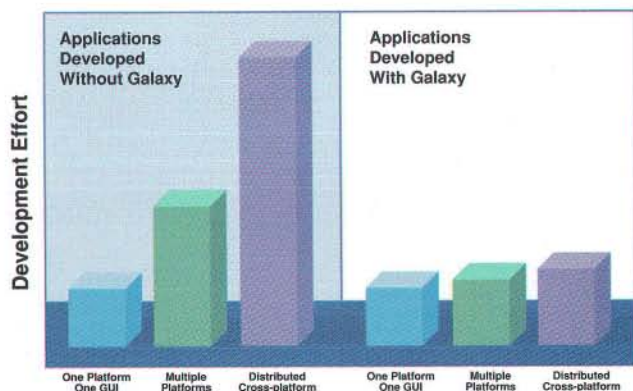
Only Galaxy offers the high-end solution you need to build mission-critical applications that scale across departments, divisions—the global enterprise.

environment. Of course, Galaxy delivers a world-class GUI builder. But, Galaxy goes far beyond simple GUI builders to deliver a rich set of platform-independent abstractions—

Our Pain is Your Gain

In building Galaxy, we learned first-hand that “bolt-ons” don’t work. Over the past seven years, we’ve poured hundreds of engineer-years into designing and re-designing Galaxy to ensure that your Galaxy applications will be limited only by your imagination. The result: Galaxy now stands at over 1.5 million lines of efficient, stable code. Yet, Galaxy programs are typically smaller and faster than those built using native toolkits. Unbelievable? Ask the editors of *UNIX Review*, or better yet, ask Galaxy customers. Galaxy’s architecture has proven itself in hundreds of real world environments.

ARCHITECTURE



The powerful Galaxy architecture enables you to bring your mission-critical applications in on-time and under-budget.

file management, networking, memory management, and distributed services—to handle the other 50% of your project. For example, the built-in Galaxy Distributed Application Services™ component provides you with the most advanced network development environment available, supporting both heterogeneous client-server and peer-to-peer architectures.

Field-proven

From Wall Street to Silicon Valley, Galaxy has exceeded the expectations of even the most demanding developers *and* won the praises of computer industry editors and analysts. Hundreds of leading corporations and major ISVs have adopted Galaxy—J.P. Morgan, Unify, IBM, Hewlett-Packard, LEGENT, Xerox, and many others are using Galaxy *today* to build their mission-critical applications.

Get the Kit

See for yourself. We’ll send you everything you need to know to begin exploring Galaxy, including industry analysis, seminar information, and an in-depth technical overview. Call us at 800.832.8668 or 703.758.8230 today. Or, if you prefer, send your request by fax to 703.758.0233 or via email to galaxy@visix.com.



G A L A X Y
Application Environment®
LIGHT YEARS AHEAD

The Blowfish Encryption Algorithm

A fast, new algorithm for 32-bit CPUs

Bruce Schneier

Blowfish is a block-encryption algorithm designed to be fast (it encrypts data on large 32-bit microprocessors at a rate of 26 clock cycles per byte), compact (it can run in less than 5K of memory), simple (the only operations it uses are addition, XOR, and table lookup on 32-bit operands), secure (Blowfish's key length is variable and can be as long as 448 bits), and robust (unlike DES, Blowfish's security is not diminished by simple programming errors).

The Blowfish block-cipher algorithm, which encrypts data one 64-bit block at a time, is divided into key-expansion and a data-encryption parts. Key expansion converts a key of at most 448 bits into several subkey arrays totaling 4168 bytes. Data encryption consists of a simple function iterated 16 times. Each iteration, called a "round," consists of a key-dependent permutation and a key- and data-dependent substitution.

Subkeys

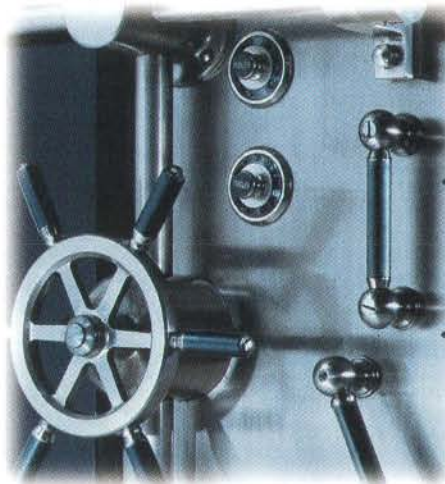
Blowfish uses a large number of subkeys that must be precomputed before any data encryption or decryption. The P-array consists of 18 32-bit subkeys, P_1, P_2, \dots, P_{18} , and there are four 32-bit S-

boxes with 256 entries each: $S_{1,0}, S_{1,1}, \dots, S_{1,255}; S_{2,0}, S_{2,1}, \dots, S_{2,255}; S_{3,0}, S_{3,1}, \dots, S_{3,255}; S_{4,0}, S_{4,1}, \dots, S_{4,255}$.

Encryption

Blowfish is a Feistel network consisting of 16 rounds; see Figure 1. The input is a 64-bit data element, x . Divide x into two 32-bit halves: x_L and x_R . Then, for $i=1$ to 16:

$x_L = x_L \text{ XOR } P_i$
 $x_R = F(x_L) \text{ XOR } x_R$
Swap x_L and x_R



After the sixteenth iteration, swap x_L and x_R to undo the last swap. Then $x_R = x_R \text{ XOR } P_{17}$ and $x_L = x_L \text{ XOR } P_{18}$. Finally, recombine x_L and x_R to get the ciphertext.

Function F looks like this: Divide x_L into four eight-bit quarters: a, b, c , and d . $F(x_L) = ((S_{1,a} + S_{2,b} \text{ mod } 2^{32}) \text{ XOR } S_{3,c}) + S_{4,d} \text{ mod } 2^{32}$; see Figure 2.

Decryption is exactly the same as encryption, except that P_1, P_2, \dots, P_{18} are used in the reverse order.

Implementations of Blowfish that require the fastest speeds should unroll the loop and ensure that all subkeys are stored in cache. For the purposes of illustration, I've implemented Blowfish in C; Listing One (page 98) is

blowfish.h, and Listing Two (page 98) is blowfish.c. A required data file is available electronically; see "Availability," page 3.

Generating the Subkeys

The subkeys are calculated using the Blowfish algorithm, as follows:

1. Initialize first the P-array and then the four S-boxes, in order, with a fixed random string. This string consists of the hexadecimal digits of π .
2. XOR P_1 with the first 32 bits of the key, XOR P_2 with the second 32 bits of the key, and so on for all bits of the key (up to P_{18}). Cycle through the key bits repeatedly until the entire P-array has been XORed.
3. Encrypt the all-zero string with the Blowfish algorithm, using the subkeys described in steps #1 and #2.
4. Replace P_1 and P_2 with the output of step #3.
5. Encrypt the all-zero string using the Blowfish algorithm with the modified subkeys.
6. Replace P_3 and P_4 with the output of step #4.
7. Continue the process, replacing all elements of the P-array and then all four S-boxes in order, with the output of the continuously changing Blowfish algorithm.

In total, 521 iterations are required to generate all required subkeys. Applications can store the subkeys rather than re-executing this derivation process.

Design Decisions

The underlying philosophy behind Blowfish is that simplicity of design yields an algorithm that is easier both to understand and to implement. Hopefully, the use of a streamlined Feistel network (the same structure used in DES, IDEA, and many other algorithms), a simple S-box substitution, and a simple P-box substitution, will minimize design flaws.

Bruce is the author of Applied Cryptography: Protocols, Algorithms, and Source Code in C (John Wiley, 1994). This article is based on a paper he presented at the Cambridge Algorithms Conference. Bruce can be contacted at schneier@chinet.com.

Write It Once.



Windows

Sell It Everywhere.



SPARC Classic

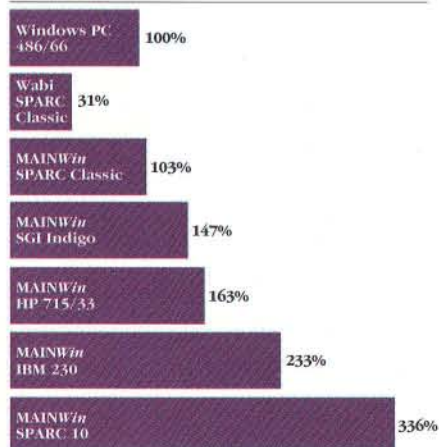
HP715/33

IBM 230

SGI Indigo

Interested in increasing your potential market by 20%, or more? Then take a close look at MainWin technology.

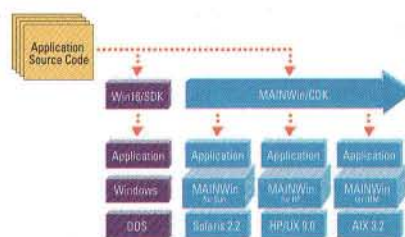
% of 486 Performance*



* Composite Code and Library benchmarks. Contact MainSoft for full details.

Far beyond the usual porting strategies, MainWin technology gives you a completely new and revolutionary cross-development paradigm. Built around a robust Windows API compatible library for UNIX, MainWin lets you leverage the Windows code base you have now into the UNIX workstation market. Not only that, in the future, MainWin will save you time, trouble and money by letting you create software products for both markets from a single code base.

And talk about performance. MainWin technology actually recompiles your Windows 3.1 code (written in C or C++ — including MFC support) to run as native code in whatever environment you've targeted. So your new UNIX applications take full advantage of their new RISC-based environments.



To find out more, call MainSoft at **1-800-MAINWIN**. And see how MainWin technology can open up profitable opportunities for you everywhere.

MAIN Soft

883 N. Shoreline Blvd.
Suite C-100
Mountain View, CA 94043

(continued from page 38)

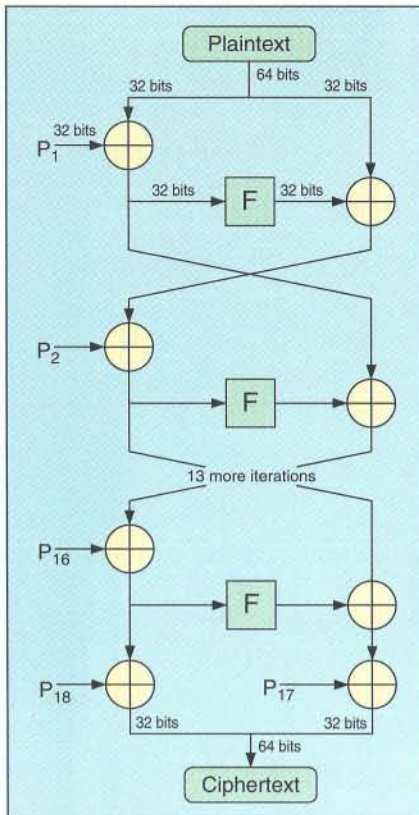


Figure 1: Blowfish is a Feistel network consisting of 16 rounds.

For details about the design decisions affecting the security of Blowfish, see "Requirements for a New Encryption Algorithm" (by B. Schneier and N. Ferguson) and "Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish)" (by B. Schneier), both to be included in *Fast Software Encryption*, to be published by Springer-Verlag later this year as part of their *Lecture Notes on Computer Science* series. The algorithm is designed to be very fast on 32-bit microprocessors. Operations are all based on a 32-bit word and are one-instruction XORs, ADDs, and MOVs. There are no branches (assuming you unravel the

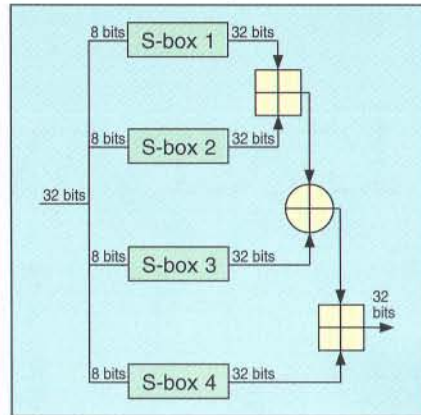


Figure 2: Blowfish function *F*.

main loop). The subkey arrays and the instructions can fit in the on-chip caches of both the Pentium and the PowerPC. Furthermore, the algorithm is designed to be resistant to poor implementation and programmer errors.

I'm considering several simplifications to the algorithm, including fewer and smaller S-boxes, fewer rounds, and on-the-fly subkey calculation.

Conclusions

At this early stage, I don't recommend implementing Blowfish in security systems. More analysis is needed. I conjecture that the most efficient way to break Blowfish is through exhaustive search of the key space. I encourage all cryptanalytic attacks, modifications, and improvements to the algorithm.

However, remember one of the basic rules of cryptography: The inventor of an algorithm is the worst person to judge its security. I am publishing the details of Blowfish so that others may have a chance to analyze it.

Blowfish is unpatented and will remain so in all countries. The algorithm is hereby placed in the public domain and can be freely used by anyone.

DDJ

(Listings begin on page 98.)

To vote for your favorite article, circle inquiry no. 4.

DDJ's Blowfish Cryptanalysis Contest

The only way to inspire confidence in a cryptographic algorithm is to let people analyze it. It is in this spirit that *DDJ* is pleased to announce the Blowfish Cryptanalysis Contest, our third reader contest in recent years.

We'd like you to cryptanalyze Bruce Schneier's Blowfish algorithm presented in this issue. Give it your best shot. Break it, beat on it, cryptanalyze it. The best attack received by April 1, 1995 wins the contest.

The contest rules are simple. It's open to any individual or organization. Governments are encouraged to enter. Even the NSA can compete and win the prize (their budget isn't what it used to be; they can probably use the money). But since we will publish the results, classified entries will not be permitted. To officially enter the contest, your entry must be accompanied by a completed and signed entry form. These are available electronically (see "Availability," page 3) or we'll be glad to mail or fax you a hardcopy.

We're not going to publish messages encrypted in Blowfish and some random key, because we think that would be too difficult.

Partial results—those attacks that don't break the algorithm but instead prove that it isn't as strong as we thought it was—are just as useful and can be entered.

Your entry does not have to consist of code. Instead, your entry can be a paper describing the attack. The attack does not have to completely break the Blowfish algorithm, it can simply be more efficient than a brute-force attack. The attack can be against either the complete algorithm or a simplified version of the algorithm (fewer rounds, smaller block size, simpler S-boxes, and the like).

We'll select a winner based on the following criteria:

- Success of the attack. How much more efficient is the attack than brute force?

- Type of attack. Is it ciphertext only, known plaintext, or chosen plaintext?
- Type of algorithm. Is the attack against full Blowfish or a simplified version of the algorithm?

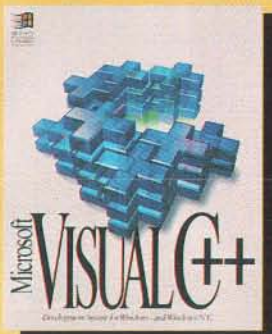
Bruce Schneier, frequent *DDJ* contributor, author of *Applied Cryptography*, and inventor of the Blowfish algorithm will referee the contest.

The contest results will be published in the September 1995 issue of *Dr. Dobb's Journal*, in which we'll discuss and summarize the winning programs, the weaknesses of the Blowfish algorithm, and any modifications of the algorithm.

We'll be providing a number of awards for the winners. The grand-prize winner will receive a \$750 honorarium. Honorariums of \$250 to the second-place winner and \$100 to the third-place winner will also be awarded.

—editors

Microsoft



Phar Lap



The Future of 32-Bit DOS!

NOW!
Supports
Borland C++ 4.0
CALL 617-661-1510

IT'S HERE!

NT power under 32-bit DOS!

Microsoft Visual C++ and Phar Lap TNT bring the never-before-available power of Windows NT to 32-bit DOS! Phar Lap's new TNT DOS-Extender lets you break the 640K DOS barrier, build multi-megabyte DOS applications and take advantage of powerful NT features. Implement threads, DLLs and multitasking with your familiar Microsoft development tools — under DOS! It's never been so easy to create the most powerful, full-featured DOS programs ever.

TNT DOS-Extender is the new standard in 32-bit DOS. The DOS power you've been waiting for is finally here!

Try it out... FREE!

If you have the Microsoft Visual C++ 32-Bit Edition tools, you've already got a free trial-size version of TNT DOS-Extender. TNT DOS-Extender



Lite is automatically installed with your Visual C++ 32-Bit Edition software. You can use TNT DOS-Extender Lite to build versatile 32-bit programs that can access up to two megabytes of memory and run under DOS, Windows 3.1, or Windows NT. It's the easiest introduction you'll find to the power of TNT.

The next-generation DOS extender.

TNT DOS-Extender is the only DOS extender to support the Win32 API, allowing your native Windows NT character-based programs to run under DOS with no changes. Programs can access all available memory — up to four gigabytes — and run with 32-bit speed and power.

But that's not all. TNT DOS-Extender also supports powerful NT features such as dynamic link libraries (DLLs) and threads, enabling developers to build modular, responsive multi-megabyte applications. And all this power is delivered with the high standards of technical excellence you've come to expect from Phar Lap, the industry leader in DOS extender technology.

Already a standard.

TNT DOS-Extender is the tool chosen by Microsoft to develop their own 32-bit tools. TNT DOS-Extender was used to build both the 16-bit and 32-bit versions of Microsoft Visual C++, Microsoft MASM 6.1 and Microsoft FORTRAN PowerStation.

32-bit CodeView, too!

TNT DOS-Extender includes a 32-bit version of the familiar Microsoft CodeView debugger. So you can use industry standard Microsoft tools, including CodeView, to develop software for the operating system of the future — that your DOS customers can use today!

TNT DOS-Extender SDK is the latest release of Phar Lap's award-winning 386DOS-Extender SDK. You can also use TNT DOS-Extender with a wide variety of 32-bit compilers (including Visual C++ 32-Bit Edition) to build Extended-DOS programs with no NT system required. In addition, TNT DOS-Extender is compatible with all 32-bit tools supported by 386DOS-Extender. An add-on run-time kit is available for developers who want to distribute TNT DOS-Extender applications to customers.

So if you've been wondering what the future holds for DOS developers, don't wait.... bring cutting-edge technology to your DOS applications today with TNT DOS-Extender!

NT Power Under DOS Lets You:

- Utilize NT features such as multitasking, DLLs and threads
- Build multi-megabyte 32-bit DOS programs
- Break the 640K DOS barrier — with your familiar Microsoft tools
- Build one application that runs under both Windows NT and DOS
- Save RAM! Run a 4 MB DOS system, not a 20 MB NT system
- Use industry-leading, high-quality Phar Lap and Microsoft tools



Phar Lap Software, Inc.

60 Aberdeen Avenue, Cambridge, MA 02138 617-661-1510 FAX 617-876-2972

386DOS-Extender and TNT DOS-Extender are trademarks and Phar Lap is a registered trademark of Phar Lap Software, Inc. Visual C++, Win32, Windows and Windows NT are trademarks and CodeView and Microsoft are registered trademarks of Microsoft Corporation. Other product and company names are trademarks or registered trademarks of their respective holders.

CIRCLE NO. 186 ON READER SERVICE CARD

WINDOWS

MAC



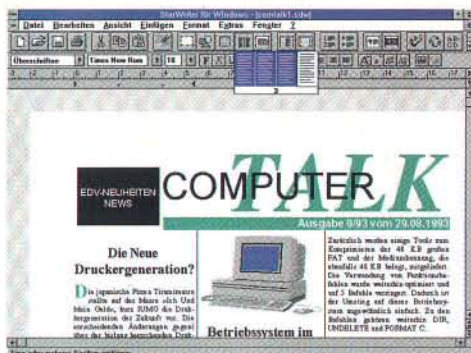
For more information on the only C++ application framework that you need to know about for porting your



UNIX

OS/2





YOU ONLY HAVE TO WRITE IT ONCE WITH STARVIEW 2.0

By learning StarView™, you can write high-end applications with a single source code for all of our supported platforms with their native look and feel and only have to debug them once—cutting your development time dramatically. StarView can guarantee seamless portability because it was designed with a multi-platform approach from the very beginning, and it has always supported the latest features—necessary for competitive high-end applications.

StarView 2.0 Features

StarView has all the standard controls like Windows, Dialogs, Menus, and Buttons. But it also has other portable capabilities like Help System, Drag & Drop, OLE, Bitmaps, Toolbar, Statusline, Tablecontrol, MDI, Printing and Preview, Internationalizing of Date, Time and Metric Classes, and a Multinational Resource System that allows you to create a single Resource file containing all international versions of your program. StarView also has standard dialogs for File-Handling, Printer-Setup, and Color- and Font-Selection.

All controls are also available in the 3D StarView Look. And in addition to the program features, StarView comes standard with the StarView Design Editor and the StarView SFX Application Framework.

applications to Windows, Windows NT, Macintosh, UNIX, and OS/2 environments, call 1-800-888-8527.

Thousands of Programmers Prefer StarView

We've developed the most complete and clearly implemented C++ multi-platform software development tools available. That's why, even after trying other libraries, thousands of programmers still prefer using StarView—from the smallest startup to the largest computer communications and systems integration companies in the U.S.

Call 800-888-8527 for More Information

If you're interested, call us to receive a full working democopy of StarView, with DesignEd, and the StarView Whitepaper. You'll be able to explore the features of StarView as well as discover how easy it is to use.

Star Division Corporation

2180 Sand Hill Road, Suite 320
Menlo Park, CA 94025
Tel: 415.233.0140
Fax: 415.233.0142
E-mail: svinfo@stardiv.de

Pricelist

Windows 3.1	\$498
Apple Macintosh	\$498
IBM OS/2 2.1	\$498
Windows NT/Intel	\$498
Windows NT/Dec Alpha	\$698
Sun SPARC Solaris 2.x/Motif	\$1498
IBM RS 6000/Motif	\$1498
DEC Alpha OSF/1	\$1498

Call for information on other platforms.



STARDIVISION™

CIRCLE NO. 782 ON READER SERVICE CARD

The Wavelet Packet Transform

Extending the wavelet transform

Mac A. Cody

The wavelet transform enables analysis of data at multiple levels of resolution (also known as "scale"). In addition, transient events in the data are preserved by the analysis. When the wavelet transform (WT) is applied to a signal in the time domain, the result is a two-dimensional, time-scale domain analysis of the signal. The transform has proven useful for the compression and analysis of signals and images.

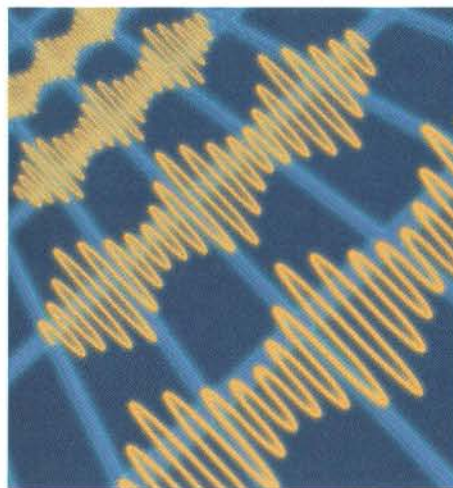
The fast wavelet transform (FWT) is an efficient implementation of the discrete wavelet transform (DWT). The DWT is the WT as applied to a regularly sampled data sequence. The transform of the data exhibits discrete steps in time on one axis, and discrete steps of resolution on another. The algorithm and a C language implementation of the FWT were presented in my article, "The Fast Wavelet Transform" (*DDJ*, April 1992).

As demonstrated in that article, the superiority of the DWT over the discrete Fourier transform (DFT) is in the DWT's simultaneous localization of frequency and time, something that DFTs can't do. As a trade-off, the frequency divisions in the DWT are not in integral steps. Instead, the divisions are in octave bands. Each level of the transform represents a frequency range half as wide as that of the level above it and twice as wide as that of the level below it; see Figure 1(a).

Mac is an engineering specialist at E-Systems' Garland Division in Dallas, Texas. He can be contacted at 214-205-6452 and on Internet at mcody@aol.com.

Conversely, the time scale on each level is twice that of the level below it and half that of the level above it; see Figure 1(b). This characteristic of the DWT poses problems when attempting to localize higher frequencies. Discrimination of frequency is sacrificed for time localization at the higher levels in the transform.

It turns out that the DWT is actually a subset of a far more versatile transform, the wavelet packet transform (WPT). Developed by Dr. Ronald A. Coifman of Yale University, the WPT



generalizes the time-frequency analysis of the wavelet transform. It yields a family of orthonormal transform bases of which the wavelet transform basis is but one member.

In this article I'll develop the wavelet packet transform algorithm from its roots in the wavelet transform algorithm. After that, I'll present C code to implement the algorithm.

From a Humble Root a Tree Shall Grow

In the fast wavelet transform algorithm, the sampled data set is passed through the scaling and wavelet filters (the convolution operation). They are, respectively, low-pass and high-pass filters with complementary bandwidths, also

known as a quadrature mirror filter (QMF) pair. The outputs of both filters are decimated (desampled) by a factor of two. The high-pass filtered data set is the wavelet transform detail coefficients at that level of scale of the transform. The low-pass filtered data set is the approximation coefficients at that level of scale. Due to the decimation, both sets of coefficients have half as many elements as the original data set.

The approximation coefficients can now be used as the sampled data input for another pair of wavelet filters, identical to the first pair, generating another set of detail and approximation coefficients at the next-lower level of scale. This process can continue until the limit for the unit interval is reached. For example, if it is desired that the transform have six levels (5 through 0), then the unit interval must be $64 (2^6)$ samples long. The data set can be of any length as long as it has an integral number of unit intervals. The resulting algorithm is the forward fast wavelet transform tree algorithm; see Figure 2(a).

You can turn the tree algorithm on end, with the initial data input at the top and the detail and approximation coefficients fanning out towards the bottom. The fast wavelet transform algorithm can now be viewed as a partial graph of a binary tree (the significance of this will be seen shortly; see Figure 2(b). The flow of the algorithm moves down and to the left, forming new levels of the transform from the approximation coefficients at higher levels. The detail "branches" are not used for further calculations.

Observe that the wavelet transform operation can be stopped at any level while working down the tree. The resulting "partial" transform is still a valid orthonormal transform. For example, if the unit interval for a data set were 32 points (2^5), the corresponding transform would have five levels (4 through 0). If the transform operation were

stopped at level 2, the transform would have only three levels, but the approximation and detail coefficients of the transform would correspond exactly to a wavelet transform with a unit interval of 8 (2^3) samples; see Figure 2(b).

The implication of this observation is that the QMF pair is an orthonormal transform kernel, just as butterfly operation is the kernel of the FFT. As long as the filters are designed to be orthonormal wavelet filters and the original data set meets the unit-interval requirement described above, repeated applications of the kernel will always yield an orthonormal transform.

Now, the set of detail and approximation coefficients at each level of the transform forms a pair of subspaces of the approximation coefficients of the next-higher level of scale and, ultimately, of the original data set. The subspaces created by the wavelet transform roughly correspond to the frequency subbands shown in Figure 1(a). These subspaces form a disjoint cover of the frequency space of the original data set. In other words, the subspaces have no elements in common, and the union of the frequency subbands span the frequency range of the original data set.

What Coifman proposed is that any set of subspaces which are a disjoint cover of the original data set is an orthonormal basis. The wavelet transform basis is then but one of a family of orthonormal bases with different subband intervals. As with the wavelet transform basis, each disjoint cover roughly corresponds to a covering of the frequency space of the original signal. Coifman dubbed this family a "wavelet packet library." The various orthonormal bases are formed by arbitrary applications of the orthonormal transform kernel upon the detail coefficients as well as the approximation coefficients of higher transform levels.

The application of the transform kernel to both the detail and approximation coefficients results in an expansion of the structure of the fast wavelet transform tree algorithm. The tree algorithm for the wavelet packet transform can be represented as a full binary tree; see Figure 3. As read from left to right, the *a* and *d* symbols at each node indicate the order of orthonormal transform kernel filter operations performed which yield each particular subspace of the original data set. Each node in the transform tree is also representative of a particular wavelet packet. The transform coefficients computed at each node are a correlation of the original data set and a waveform function representing the wavelet packet.

For example, the sequence *aaad* in Figure 3 represents four operations of the orthonormal transform kernel representing one of 48 possible wavelet packets. (The combination of all possible translations in time and dilation in scale for the wavelet packets is $J \cdot 2^J$; in this instance J equals 4.) The first three represent low-pass filter/decimation operations performed by the transform kernel. The fourth represents a high-pass filter/decimation operation performed by the transform kernel. This subspace should be recognizable as exactly the level 0 detail coefficients of

the wavelet transform. The operations of the orthonormal transform kernel correspond to the wavelet function of the wavelet transform. Likewise, the wavelet packet represented by *aaaa* is the scaling function of the wavelet transform.

Packets, Graphs, and Bases

The wavelet transform basis is actually a subset of a family of bases formed by the wavelet packet transform. The heavy lines in Figure 3 indicate the graph forming the wavelet basis. Note that the wavelet basis consists of the subspaces

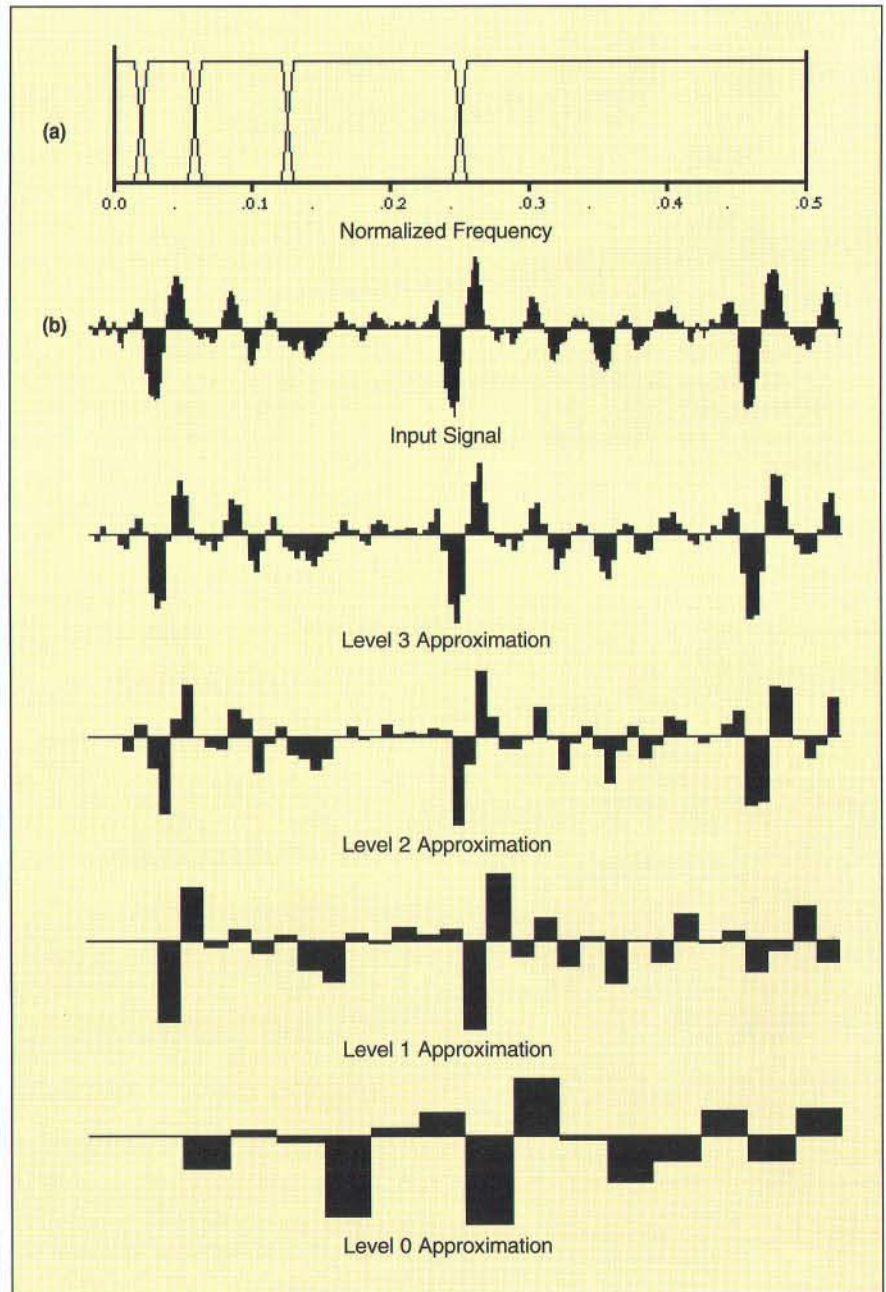


Figure 1: (a) The Discrete Wavelet Transform (DWT) divides the spectrum of the sampled data into octave bands; (b) the resolution at each level of the DWT is half that of the level above it and twice that of the level below. At lower levels, time resolution is sacrificed for frequency localization.

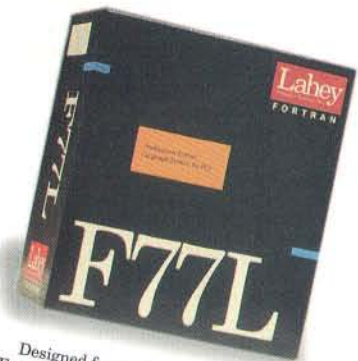
Just Add Code.

See Us at Software Development Booth #1913

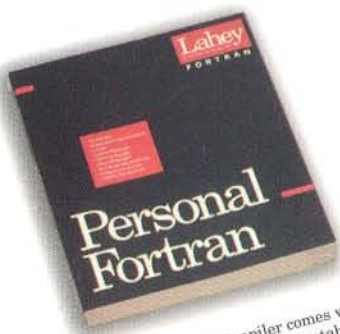


Award-winning 32-bit Fortran language system for writing or downsizing Fortran applications as large as 4 GB.
\$795

Call us about our upcoming Fortran 90 Language System.



Designed for professionals writing Fortran programs running under DOS. This language system's tools include an editor, debugger, profiler, linker, library manager, and make utility.
\$295



This fast ANSI 77 compiler comes with an editor and debugger at an unbeatable price.
\$99

(800) 548-4778

Free, Unlimited Technical Support
30-Day Money-Back Guarantee
Same-Day Shipping



Fortran is our forte

(702) 831-2500
Fax: (702) 831-8123
P.O. Box 6091
Incline Village NV 89450

CIRCLE NO. 262 ON READER SERVICE CARD

d , ad , aad , $aaad$, and $aaaa$. The sequences a , aa , and aaa are intermediate steps leading to the generation of the subspaces of the wavelet basis at the lower levels. Since the orthonormal transform kernel can be arbitrarily applied to either approximation or detail branches on the tree, $J \cdot 2^J$ graphs rep-

resenting different orthonormal bases can be created; see Figure 4.

The variety of orthonormal bases which can be formed by the WPT algorithm, coupled with the infinite number of wavelet and scaling functions which can be created, yields a very flex-

(continued on page 50)

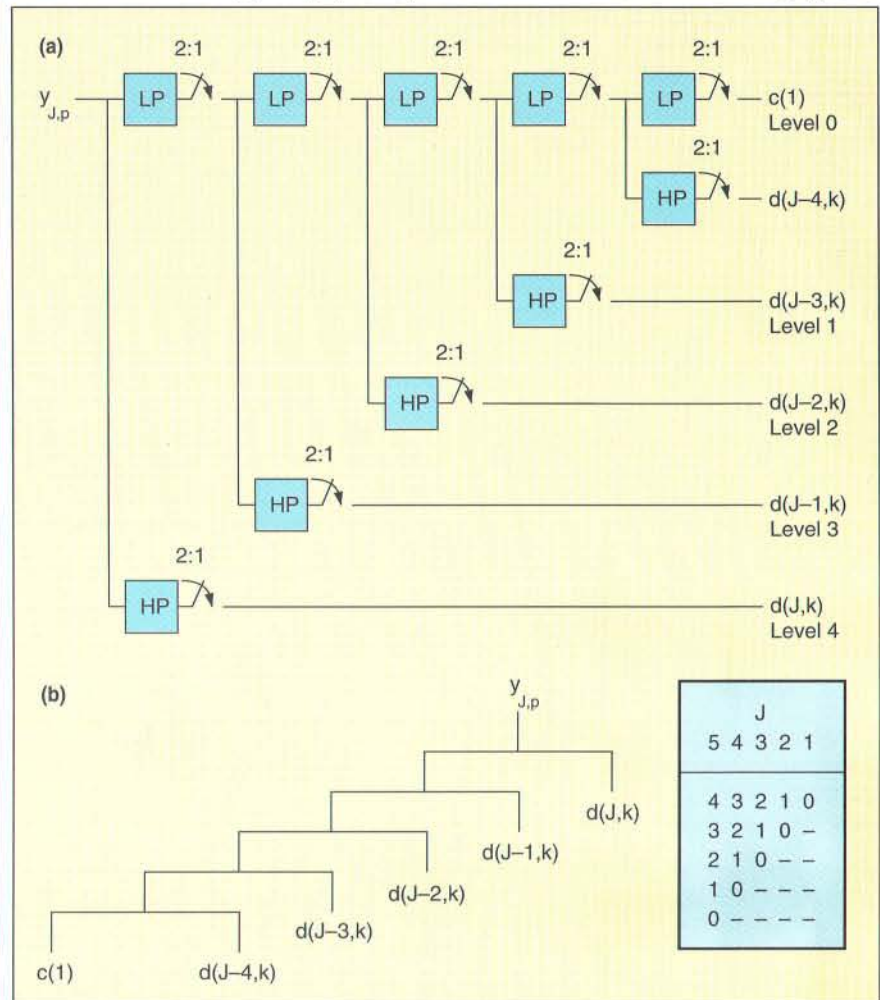


Figure 2: The tree or pyramid algorithm of the forward fast wavelet transform (a) can be viewed as a partial graph of a binary tree (b). For a particular unit interval (2^J samples), a maximum of J levels of transform data can be formed.

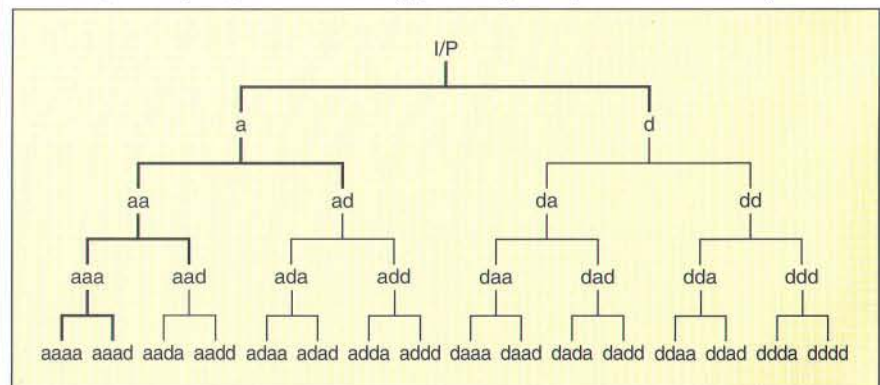


Figure 3: The wavelet packet transform viewed as a complete binary graph. Each "a" and "d" in each sequence represents the filtering operations performed to yield the particular subspace of the original signal. The bold lines represent the disjoint cover known as the "wavelet basis."



Imagine What You Could Do With The Time Our Software Publishing Services Will Save You.

Let's face it. You could use a little time away from the office. All you have to do is make Digital your software publishing partner. We have state-of-the-art ISO 9000 manufacturing facilities around the world. We're equipped to handle virtually any medium. And we have over thirty years of experience in media and format conversions, documentation, package design, kitting, language translation and worldwide distribution.

Leave it to us, and you'll have more time to do the things you like best. And that's no fish story. Call us for a free catalog.

1-800-448-6737

Putting Imagination To Work

digital

POWERBUILDER DEVELOPERS WANTED

We're looking for senior application developers and technical architects to join our rapidly expanding client/server team. To succeed, you will have relevant experience which must include at least 6 months of PowerBuilder development within an SQL compliant relational database environment.

Wellington Associa
Fax: 800-555-83

Looking
for

CLIENT/SERVER EXPERIENCE

Unique opportunities are available for programmers experienced in implementing Client/Server systems. Applicants must have a four-year technical degree, four to six years' development experience, and proficiency with both Oracle and PowerBuilder. Benefits include tuition.

PROGRAMMERS/ANALYSTS

Information consulting firm has opportunities throughout the East as well as in California for professionals with strong technical backgrounds in system design, development and methodology, as well as experience with Windows, DB2 and PowerBuilder. Benefits include excellent salary, continuing education, and much more.

ment Services

POWERBUILDER PROGRAMMERS

Looking for seasoned PowerBuilder programmers for a major development effort. Environment includes Novell LAN, SQL Server running on NT. Interested primarily in independent developers but will consider companies. Need is urgent. Please fax interests and background to (800) 555-2222.

Computer Network
800-555-2222

ANALYSTS/PROGRAMMERS

PowerBuilder P/A's	OPEN
ORACLE/SQL*Plus	50-60k
SMALL TALK	40-60k
HOGAN P/A's	45-65k
IMS/DB2/DMS	40-50k
EDP Auditors	40-50k
UNIX/C/C++/ADP	40-50k
IDEAL/DATACOM	40-50k
ORACLE V6.0 & 7.0	40-50k
AS/400/SP's P/A's	40-50k
MVS/CICS/COBOL	35-45k

For additional information on these and nationwide opportunities, fax resumes in confidence to:

CAREERS
UNLIMITED
800-555-3999

WARE APPLICATION MANAGER

Company needs professional to manage a team of dBASE working on client/server applications for end users in our administration department. Candidates should have 4 plus years management experience, expertise in database application development, and proficiency in PowerBuilder.

HARTMAN & COMPANY
800-555-6318

Credentials for sale: \$249.



When it comes to developing client/server applications, the success of PowerBuilder™ is evident simply by glancing at the front pages of almost any trade journal. *InfoWorld* and *LAN Magazine* both called it "Product of the Year." *DBMS* gave it two Readers' Choice Awards. And *Windows Sources* honored it with its Experts' Pick Award.

But PowerBuilder's popularity is just as obvious when you peruse the back pages of a publication. For there you can see how many companies need programmers with experience in the industry's most proven client/server development tool.

So you'll be glad to know that PowerBuilder, previously available to developers with corporate IS needs and resources, is now available to desktop database developers as well.

Thanks to new PowerBuilder Desktop. A version of PowerBuilder created specifically for your standalone and networked desktop-class databases.

Similar to PowerBuilder Enterprise, this edition lets you create applications using a common object technology.

A technology that lets your programs, as well as your programming know-how, be scaled to run against any database from .DBF to DB2®.

With hundreds of built-in functions, and tools that include our "SQL Smart" DataWindow™ and a built-in data dictionary, you can build those applications with a simplicity that's unmatched in a Windows® environment. And, with the full 32-bit relational power of the WATCOM™ SQL database, you can build and deploy standalone applications right out of the box.

If that's not appealing enough, for a limited time, you can get PowerBuilder Desktop at the introductory price of \$249. Simply call Powersoft at 1-800-866-8623, your corporate reseller or stop by CompUSA. Or call us at 1-800-946-3500 and we'll tell you the nearest location and date of our one-day comprehensive client/server training class that includes your own copy of PowerBuilder Desktop for just \$399.

After all, the price of becoming a client/server developer is a lot lower than the price of not becoming one.

Powersoft™
Building on the power of people.

Powersoft Corporation, 70 Blanchard Road, Burlington, MA 01803
Powersoft Europe, Thames House, 1 Bell Street, Maidenhead, Berkshire, SL6 1BU, United Kingdom

All trademarks and registered trademarks are property of their respective owners. Prices listed do not include sales tax, shipping and handling. 30-day money back guarantee.

CIRCLE NO. 922 ON READER SERVICE CARD

SLATE with Graphics

SLATE

SCRIPT & S_PRINT

Would Text and Graphic Printer support for over 850 printers give you an edge?

By including **SLATE with Graphics**, you can print Text and Graphics on over 850 printers. **Immediately! Painlessly!**

You can use **SLATE** in your product with **no royalties**. It gets you **out of the printer support business**.

Make your product more functional and competitive by using SLATE's advanced text features:

- Output to parallel printers, serial printers, DOS files and Novell network printers.
- Support proportional fonts and scalable fonts.
- Set exact print positions.
- Kerning, leading, underlining, and strike through.
- Automatic character set conversion.
- Print lines and shaded areas (laser printers only).

SLATE with Graphics adds advanced graphic printing features:

- Print images from the screen, PCX or TIFF files, or custom image systems.
- Print lines and shaded areas on all printers.
- Scale and Rotate printed image.
- Print grey scale and color images.
- Intermix text and graphics.

SLATE is a **C** or **Basic** library of over 170 text printing functions, a **Database** of over 850 printers, and **End User configuration and testing** programs. **SLATE with Graphics** adds over 60 graphic printing functions.

Would a User Configurable Report Writer give you a more competitive product?

SCRIPT is a **full featured Text Formatting library** that can be incorporated into your application. **SCRIPT** uses **SLATE** as its printer driver. **SCRIPT** lets you merge text, data, and **S_PRINT** formatting commands from ASCII files and your application.

Enhance your product by taking advantage of SCRIPT's features:

- Allow users to alter document format.
- Set exact positions, center, right adjust, and decimal align.
- Fill paragraphs from unformatted text.
- Add lines, shaded areas, logos, signatures, etc.
- Add commands and macros.

Call or FAX now for a complete catalog and developer's guide for The Symmetry Group's printer support products. **Order SLATE for \$299, SLATE with Graphics for \$448, or SCRIPT for \$199** with our risk free, 30 day return policy.

The Symmetry Group
800-346-3938
PO Box 26195
Columbus, OH 43226, USA
614-431-2667 • FAX 614-431-5734

(continued from page 46)

ible analysis tool. The flexibility of WPT versus the FWT can be compared to that of having a complete set of sockets for a ratchet rather than a single socket to attach to it. The ratchet (algorithm) works the same regardless of the socket (basis) that is chosen. The flexibility of the tool is in choosing the appropriate socket (basis) for the particular nut (problem). The choice of wavelet and scaling functions is then analogous to selecting from English, metric, or Torx socket sets for use with the ratchet. The WPT allows tailoring of the wavelet analysis to selectively localize spectral bands in the input data as well as to correlate the signal to the wavelet. Not only can the best wavelet be chosen to analyze a particular signal but the best orthonormal basis can as well. In signal-processing terminology, the various bases of the wavelet packet transform can be used as arbitrary adaptive tree-structured filter banks.

Piece-wise Convolutions and Traversing the Tree

The implementation of the WPT is itself a generalization of the FWT routine presented in my previous article. As with the FWT, the kernel operations are the decimating and interpolating convolutions, as presented in Listing One (page 101). The convolutions performed are actually piece-wise convolutions, due to the discrete nature of the data. The routine *DualConvDec2* replaces the routines *ConvolveDec2* and *Dotp* in the FWT code. The routine *DualConvInt2Sum* replaces *ConvolveInt2*, *DotpOdd*, and *DotpEven* in the inverse FWT code.

Both convolution routines are designed to operate upon aperiodic data of finite length. The data does not represent an infinitely repeating pattern and is assumed to be surrounded by zero-valued data; see Figure 5. To support this data model, each data array

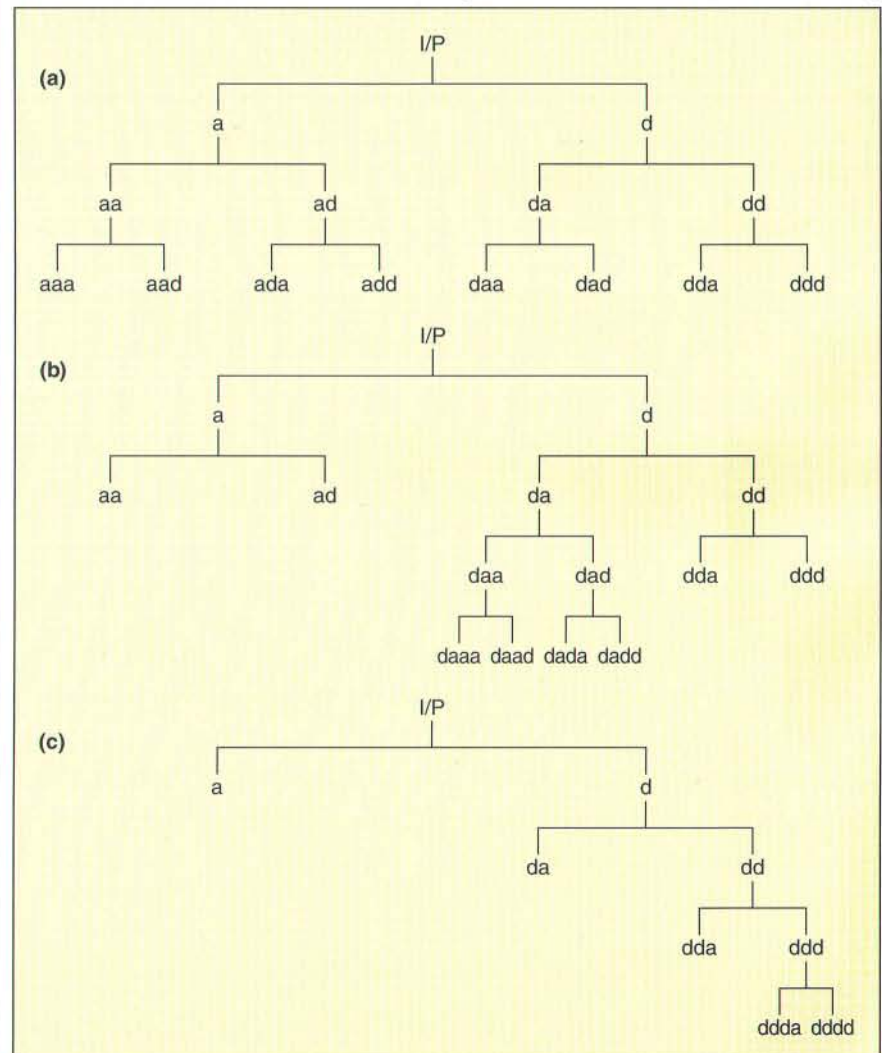


Figure 4: Different disjoint covers formed from the WPT binary tree (Figure 3) yield different wavelet packet bases. (a) A subband basis; (b) an orthonormal basis subset; (c) a basis which is the opposite of the wavelet basis (better frequency localization at higher frequencies).

Introducing IBM C Set ++ V2.1 and KASE:Set.

Object oriented development.



C Set ++ Version 2.1 for OS/2® from IBM Software Solutions is one of the most complete object-oriented application development packages you can buy.

Plus.

Its 32 bit C/C++ compiler, with its advanced code optimizer *and*

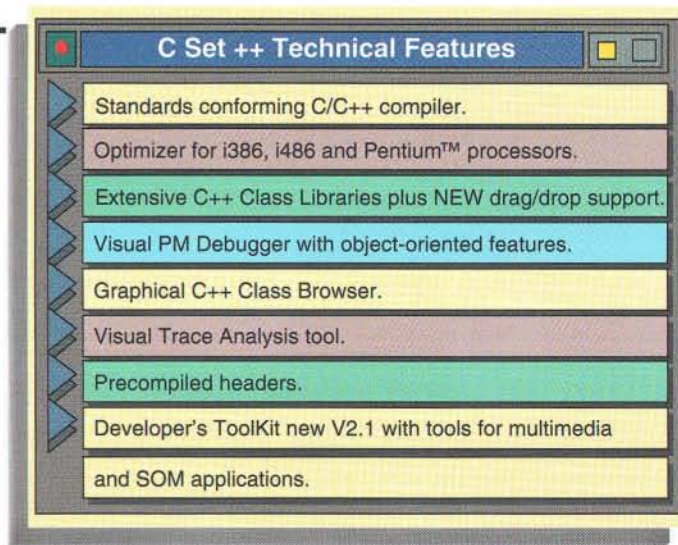
WorkFrame/2 V2.1 - a

completely new development environment based on

C Set ++ OS/2's object-oriented Workplace Shell - allows you to create up-to-the-minute mission critical applications.

You also receive a specially tailored copy of Kaseworks KASE:Set, a visual design and code generation tool which allows you to build visual applications even more quickly, with a minimal investment in learning time.

Plus!



To order C Set ++ for OS/2 including KASE:Set, or for further information call 1-800-342-6672 (U.S.A.) or 1-800-465-7999, ext. 676 (Canada). Or contact your local IBM software dealer.

C Set ++ V2.1
including Kaseworks KASE:Set.
Starting at \$393 (U.S.) - CD-ROM Version.

(continued from page 50)

is appended with additional data storage equal to the length of the wavelet filter, minus one (the shaded elements). The extra data is filled with the terminating convolution values as the wavelet filters "slide off" the end of the data set. Extending the convolution data by this additional amount during the decomposition process of the forward transform ensures perfect reconstruction of the original input data by the

The discrete wavelet transform is actually a subset of a far more versatile transform, the wavelet packet transform

inverse transform. *DualConvDec2* also uses partial dot products at both ends of the convolution to simulate the implied zero-valued data (the dotted lines) outside of the data array. The dotted lines on the filter elements indicate coefficients not used in the partial dot product calculations. *DualConvInt2Sum* does not need to do this, since all information necessary for reconstruction is contained within the extended data arrays. Note that *DualConvInt2Sum* performs only the odd-valued dot product at the beginning of the convolution for the initial, reconstruction data point.

The WPT data is stored in the structure *WPTstruct*, defined in Figure 6(a). The structure contains storage for the number of levels in the transform, the length of the original, untransformed data array, and a pointer to a two-dimensional matrix of data arrays. The size of the matrix is dependent upon three factors. These are the number of levels in the transform, the length of the original data array, and the length of the transform filters. The length of the data array is itself affected by the length of the transform filters; see Figure 6(b).

Figure 6(c) shows the matrix structure as the wavelet packet binary tree. The data-array pointers are allocated memory from the heap as necessary to form the disjoint cover for the chosen orthonormal basis. Those array point-

ers not required for the disjoint cover are set to zero. The example shown in the figure represents a three-level wavelet basis for an input of 40 data points with transform filters containing six coefficients.

The routine *DualConvDec2* is used by *AWPT*, the forward wavelet packet transform routine; see Listing Two (page 101). *AWPT* accepts pointers to the input data array, the WPT data structure, and the transform filter arrays, and the length of the filters. The transform routine works down the levels of the binary tree performing convolutions on the data arrays. Each level of the binary tree is traversed by taking adjacent pairs of array pointers as the destination nodes for the low-pass and high-pass convolutions. If the array pointer for the low-pass convolution is zero, the convolutions are not performed since the destinations are not part of the current disjoint cover.

At the highest level (where i equals 0) the input data array is the source

for the convolutions. On each subsequent level, the sources are the arrays on the previous level. The appropriate array in the binary tree is determined by dividing the current j index by 2. After each convolution operation, the data length is divided by two, in order to keep track of the effect of the decimation operation during the convolutions.

DualConvInt2Sum is used by *IWPT*, the inverse wavelet packet transform routine. *IWPT* accepts pointers to the source WPT data structure, the output data array, the transform filter arrays, and the length of the filters. The inverse transform routine works up the levels of the binary tree, performing convolutions on the data arrays and reconstructing the higher-resolution data on each level. Each level of the binary tree is traversed in the same fashion as was *AWPT*. The destination arrays are on the next-higher level of the matrix, and they are selected by dividing the j index by 2. At the highest

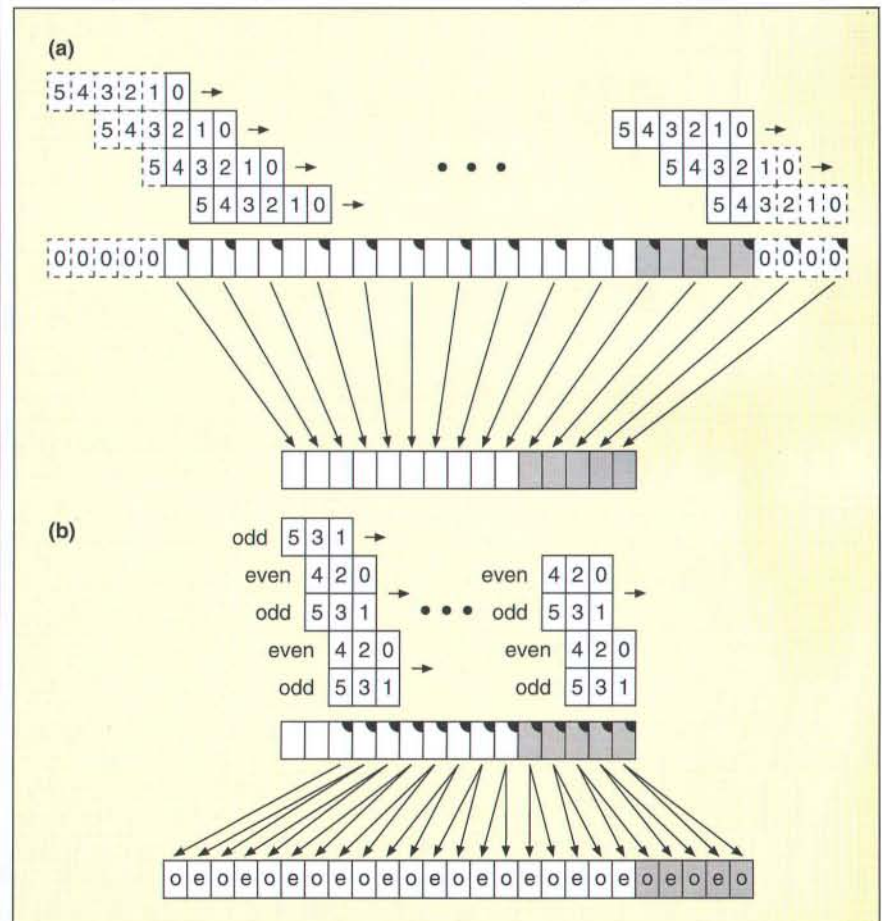


Figure 5: (a) The convolution operation in *DualConvDec2* employs partial dot products at both ends of the data array to simulate zero-valued data surrounding it; (b) the convolution operating in *DualConvInt2Sum* performs dot products with alternating odd and even filter components to simulate interpolation. The additional convolution data generated by *DualConvDec2* is used to accomplish perfect reconstruction. Marked data elements indicate starting points of dot-product calculations.

Windows & DOS Programming Tools

Sourcer™

New
v5.0

"Sourcer is the best disassembler
we've ever seen." PC Magazine

Creates commented source code and listings from binary files. Shows how programs work with detailed comments on interrupts, subfunctions, I/O functions, and more. Supports all instructions to 80486 and V20/V30.

Sourcer provides the best analysis separating code and data. It automatically determines data types, uses descriptive labels for BIOS and PSP data, and links data items across multiple segments.

New version 5.0 makes most DOS EXE and COM files and drivers reassemble perfectly, byte-for-byte identical to the original!

Top professionals depend on Sourcer for the most reliable results with the least effort.

New
v2.0 for Windows

"Sourcer combined with Windows
Source should be mandatory for
looking into Windows Programs."
Sal Ricciardi PC Magazine

Windows Source™ with Sourcer generates detailed listings of Windows EXEs, DLLs, SYSs, VxDs, device drivers & OS/2 NE files. Labels, by name, export & import function calls, API calls like "GetFreeSpace" and more.

See the many undocumented Windows functions used by professionals to perform tricks that are otherwise impossible.

Comes complete with extra utilities for resource extraction and import analysis. Uses CodeView symbols for improved clarity.

BIOS Source

for PS/2, AT, XT, PC and Clones

The BIOS Pre-Processor™ with Sourcer creates commented listings for any BIOS ROM in your PC. Understand how your specific BIOS works! Adds over 75K of comments specific to your BIOS. Identifies multiple interrupt branches with special labeling like "int_10_video." Fully automatic.

Sourcer -Commenting Disassembler	\$129.95
Sourcer w/BIOS -(save \$10)	169.95
ASMtool 486 -Automatic flowcharter	199.95
ASM Checker -Finds source code bugs	99.95
Windows Source -requires Sourcer	129.95
Windows Source & Sourcer -(save \$30)	229.90

Shipping: USA \$6; Canada/Mexico \$10; Other \$18, CA residents add sales tax. © 1993 VISA/MasterCard/COD

30-DAY MONEY-BACK GUARANTEE

1-800-648-8266 order desk

V Communications, Inc.

4320 Stevens Creek Blvd., Suite 275-DD
San Jose, CA 95129 FAX 408-296-4441
408-296-4224

(continued from page 52)

level (i equals 0), the destination array is the output data array. After each convolution operation, the data length is doubled to keep track of the effect of the interpolation operation during the convolutions.

The code listings presented here are written in ANSI C and have been tested with Borland Turbo C 2.0. They should compile on any compiler that is compliant with ANSI C. I've also written a wavelet packet transform demonstration program which is available electronically; see "Availability" on page 3. The electronic version includes the demo program, sample data files, support drivers, and documentation.

Conclusion

The wavelet packet transform generalizes the discrete wavelet transform and

provides a more flexible tool for the time-scale analysis of data. All of the advantages of the fast wavelet transform are retained since the wavelet basis is in the repertoire of bases available with the wavelet packet transform. Given this, the wavelet packet transform may eventually become a standard tool in signal processing.

References

Cody, Mac A. "The Fast Wavelet Transform." *Dr. Dobb's Journal* (April 1992).

Coifman, Ronald R., Yves Meyer, and Victor Wickerhauser. *Wavelet Analysis and Signal Processing*. New Haven, CT: Yale University, 1991, preprint.

DDJ

(Listings begin on page 100.)

To vote for your favorite article, circle inquiry no. 5.

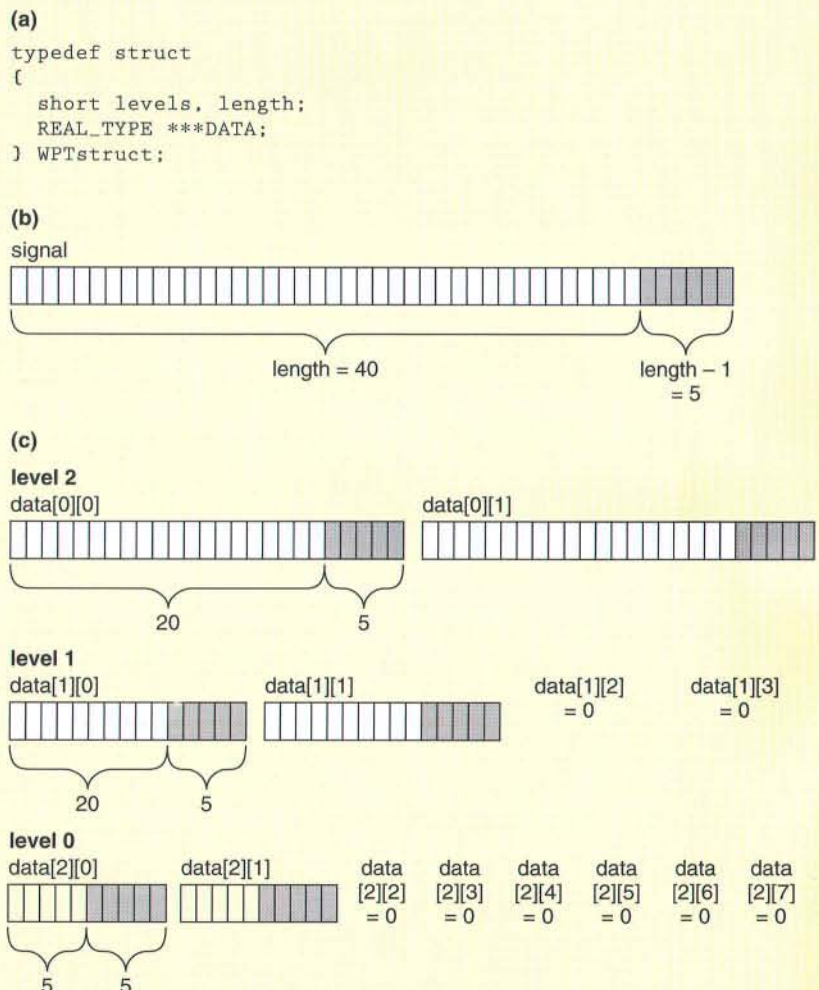


Figure 6: (a) The data structure for the WPT. The REAL_TYPE declaration can be defined either as float or double; (b) the storage structure for the original signal consists of the data, plus appended storage equal to the filter length, minus 1; (c) the data component of WPTstruct is a two-dimensional matrix of data arrays forming the wavelet packet binary tree. Data-array pointers set to 0 indicate parts of the tree that aren't part of the disjoint cover (the wavelet basis).

Microsoft Press Programming Series

DISKS INCLUDED

INSIDE OLE 2

The Fast Track
to Building
Powerful
Object-Oriented
Applications
with Windows™
Objects



KRAIG BROCKSCHMIDT



ISBN: 1-55615-618-9, two 3.5-inch disks, \$49.95

Objectivity

It's a giant leap into the world of object-oriented application development for the Microsoft® Windows™ operating system.

And we do mean giant. You get more than 1,000 pages of expert advice from one of the best programming minds. Plus, over 50,000 lines of practical source code examples in C/C++ on two

high-density disks let you see OLE 2 programming in action. You'll learn to build applications from scratch or add OLE 2 functionality incrementally for whatever makes sense for your customers, clients, or business.

The future is now with OLE 2, so get the insider's guide today. Available everywhere quality computer books are sold, or call 1-800-MSPRESS. Refer to ad ADJ.

MicrosoftPress

High-Performance Computer Books

One Microsoft Way, Redmond, WA 98052-6399
In Canada, call Macmillan Canada, 416-293-8141

Microsoft is a registered trademark and Windows is a trademark of Microsoft Corporation.

Fuzzy Logic in C: An Update

Completing a fuzzy-based inference engine

John A.R. Tucker,
Phillip E. Fraley, and
Lawrence P. Swanson

Early last year, we were looking for a software implementation of fuzzy logic. Greg Viot's article "Fuzzy Logic in C" (*DDJ*, February 1993) was a step towards what we needed, but it didn't include the necessary initialization, parsing, and output functions. Consequently, we filled in the gaps by writing functions that, together with Greg's code, make a working fuzzy-logic program you can use. Listing One (page 101) is the complete source code for the updated version (which includes Greg's original code and our additions). The enhancements, which we'll focus on in this article, are shaded as well as identified in the comments. For background on fuzzy logic in general and Greg's techniques in particular, refer to his original article.

Rule Files and Structures

We saw right away that the parsing of the rules file would be a problem to generalize for all possible combinations

John teaches computer courses at Albright College and Reading Area Community College; Phillip is working on several projects, including proton models, large color images, and neural networks; Lawrence currently works as a test engineer. They can be reached through the DDJ offices.

of antecedents and consequences, so we elected to simplify the problem by allowing only two antecedents and one consequence.

The generalized case would have resulted in loss of clarity. We didn't try to be clever about our functions; in fact, they are quite direct (three segments are repetitive). The extensive use of linked lists and pointers to structures in *initialize_system()* related to the rules are, however, quite involved. Nor did we optimize or generalize the code. This makes it possible for you to modify the code to accept other input files by copying existing code segments and making minor adjustments. Finally, we took full advantage of understanding the input data structures for the specific example of the inverted-pendulum problem Greg described.

To allow for easy alteration of the fuzzy sets or rule definitions, we used three ASCII files with fixed names and formats as the input files that describe the fuzzy sets (angle, velocity, and force). Similarly, an ASCII file is used to describe the rules file. These four files are to be located in a common directory from which the program is run.

In the three files describing the fuzzy sets (in1, in2, and out1), you can use any name ten characters or less in length on the first line as a name for the input fuzzy set. The first column of the subsequent lines is for the name of the membership element of that fuzzy set, again limited to ten characters. The next four columns describe the corner points of the membership (if the third and fourth columns are the same, the shape is a triangle). White-space, spaces, or tabs separate the columns. You may have as many rows of membership elements as you please, but five, seven, or nine seem to be the best choices. Take care not to include any blank rows.

The first file, in1 (angle), looks like Figure 1. The files in2 (velocity) and

out1 (force) are similar. In *initialize_system()*, we have three nearly identical code fragments. You can block copy them and make the few changes required. The cycle is as follows: Open the file, set a pointer and allocate memory, read the fuzzy set's name, read a line of data from the file, set a pointer to the next structure, assign values to the structure elements, and lastly, close the file. The differences in these three segments are in lines 1 and 2 (the filenames are different), lines 5 and 6 (the pointers point to differing places), and lines 27 and 33, where the filenames in the error messages are different.

We included an error trap to detect if either *slope1* or *slope2* is less than or equal to 0, a condition not allowed in the original program. If such an error is encountered, the program exits with appropriate information. The setting of the pointers for the rules file is more complex. In the original article, Greg suggested a file that looked like Figure 2(a). Although we liked the form of this file, it was complex to parse so we stripped the file to its essential elements: the name of the fuzzy set elements and the order in which they appear in each rule; see Figure 2(b). We used an awk and sed pipeline to strip the "rule" file and create a more suitable form for parsing in a file named "rules." (The command line *awk '{print \$6, \$10, \$14}' rule | sed 's/)/g'> rules* does this elegantly. You can create the rules file di-

Angle				
NL	0	31	31	63
NM	31	63	63	95
NS	63	95	95	127
ZE	95	127	127	159
PS	127	159	159	191
PM	159	191	191	223
PL	191	223	223	255

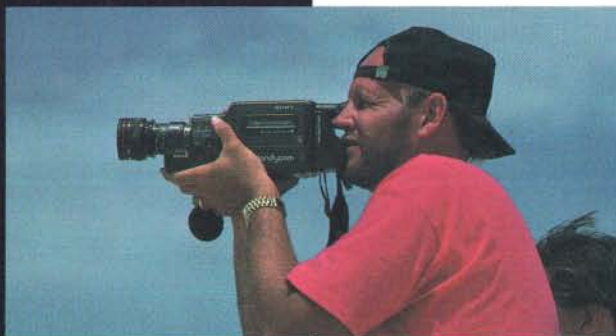
Figure 1: The in1 file; values can be altered as desired.

NEW VERSION 3.0 NEXPERT OBJECT[®]

Building Smarter Applications[™]

"NEXPERT has effectively captured the design engineer's knowledge required to perform CCD simulations, significantly improving productivity at SONY."

SONY



SHORTENED TIME TO MARKET FOR CAMCORDERS

NEXPERT is the core of SONY's simulation system for speeding up the design and testing of highly-specialized chips used in video cameras. The system, called XAS, reduced design time by two thirds. XAS runs in a local area network of Sony NEWS workstations, using X Windows, with more than 500 complex simulation rules distributed across 15 NEXPERT knowledge bases.

"By adding knowledge to existing applications, NEXPERT has allowed us to leap beyond our already strict audit control standards to help us perform a difficult and critical task."

Chemical Bank



RISK MANAGEMENT

Chemical Bank uses NEXPERT for daily review of over a billion dollars in worldwide foreign exchange transactions. The Digital VAX-based application called Inspector, ties to Oracle databases, C programs, and a communications network spanning 23 countries. Given the dollar amounts involved in transactions, once Inspector identified even one fraudulent trade, it paid for itself many times over.

"We've recognized that knowledge is a resource that can be harnessed and leveraged to strengthen productivity and reduce costs in many areas of our business — and NEXPERT is the key."

BCTel



CUSTOMER SERVICE

BCTel, Canada's second largest telephone company, uses NEXPERT to streamline business practices and help generate, recover, and protect revenue. NEXPERT is at the heart of several multiplatform applications ranging from network overload management to customer services and billing monitoring. The systems run on Sun and Digital UNIX workstations, Digital VAX/VMS systems and PCs.

BUILT ON STANDARDS

NEXPERT OBJECT is an object-and rule-oriented development tool written in C, that runs on over 35 platforms, from PCs and Macs to Unix, DEC VAX/Alpha and IBM mainframes. NEXPERT extends beyond other knowledge-based tools with functionalities such as a robust GUI design capability and a script language. Developers can now more rapidly build graphical knowledge-based applications that support all windowing environments.

With over 16,000 systems in use worldwide and an extensive network of solution providers and systems integrators, NEXPERT has emerged as the proven standard for developing knowledge-based systems.

Call us today at 415-321-4488 or 1-800-876-4900 ext. 804 to register for our free seminars or to receive additional documentation.

CIRCLE NO. 602 ON READER SERVICE CARD

*Object-Oriented Tools for Portable
Client/Server Development[™]*



(continued from page 56)

rectly, as with the input files, and eliminate the clearer representation of the rule file entirely if you do not have these tools.) You can have more or less than the 15 rules in Greg's article. Add or delete them as you please, one rule per row. Be certain that membership-element names are exact matches in all the files, including the rules file. In particular, note that upper and lower case are not equivalent.

Once *initialize_system()* is written, you're limited to two inputs and one output. You will need to make changes to the arguments for *fscanf()* and define new buffers to accommodate any other combination.

The insight to the rules structures initialization is that structures of *rule_type* and *rule_element_type* form the acceptable rules at the time of initialization. That is, appropriate fuzzy inputs (antecedents) are associated (linked) with a fuzzy output (consequence) as defined in the rules at the time the rules file is read. Values in the *mf_type* structure are pointed to by the pointer stored in the *rule_element_type *value*. Later, if a 0 is pointed to by any of the *if_side* value pointers, the function *defuzzification()* will equate to 0, and subsequent calculation of the *sum_of_products* and *sum_of_areas* will not be affected. See Figure 3 for the complete relationship of all the data structures and their pointers.

Using the Updated Program

To illustrate how you can use the updated fuzzy-logic program, we'll refer you to the rule in Figure 4, where we begin by opening the rules file, allocating memory for a structure *rule_type* and setting a pointer to it, and scanning the rules one line at a time. As each line (rule) is read, we "know" that the first field in the line is the angle (structure *io_type*, pointed to by **membership_functions*), so we begin searching its fuzzy-set members (structures *mf_type*), doing a string match on the membership element name, NL. When the match is found, memory for a *rule_element_type* structure is allocated, the *address* to the *value* element of the matching *mf_structure* is stored, and a pointer to *rule_type*, pointed to by the **if_side*. A pointer to the second field (in this case, velocity) is also established as a pointer (element **next*) in *rule_element_type*.

The second field of the rule (velocity) is then used to search for a string match on its membership-element name, ZE, in the second *io_structure *membership_function*, and a pointer to the *address* where its value is located is stored in the next *rule_element_type *value*. Finally, the last element of the rule, the

```
(a) rule 1: IF (angle is NL) AND (velocity is ZE) THEN (force is PL)
    rule 2: IF (angle is ZE) AND (velocity is NL) THEN (force is PL)
    rule 3: IF (angle is NM) AND (velocity is ZE) THEN (force is PM)
    ...
    rule 15: IF (angle is PL) AND (velocity is ZE) THEN (force is NL)

(b) NL ZE PL
    ZE NL PL
    NM ZE PM
    ...
    PL ZE NL
```

Figure 2: (a) Original rules file; (b) modified rules file.



The World's Most Scalable, Portable, and Interoperable Databases Start With "M".

**FREE
Demo Software**

M is a powerful run-time development environment designed to create scalable applications that port across *all* operating environments — simply and with minimal programming time. And since M applications are easily ported to environments like Windows, X Windows/Motif, Windows NT, UNIX or DOS, you'll want to build all kinds of open solutions around it — as Fortune 500 companies have been doing for years. So, if you need a world-class, industry-standard database, call the M Technology Association today for FREE demo software and applications catalog. Then, get started on something great.



1738 Elton Road, Suite 205, Silver Spring, MD 20903-1725 Phone: 301-431-4070 / Fax: 301-431-0017
All product and company names are trademarks of their respective companies.

CIRCLE NO. 910 ON READER SERVICE CARD

FUZZY LOGIC HAS FINALLY GOTTEN DOWN TO BUSINESS.

INTRODUCING *FUZZYSP*

The First Fuzzy Logic System Designed Specifically to Solve Business Problems.

For more information about *FuzzySP*, call (412) 787-8222 or
FAX (412) 787-8220. When you learn more about it, you'll
see why *FuzzySP* from NeuralWare can put your bottom line
into much sharper focus.



NEURALWARE

CIRCLE NO. 170 ON READER SERVICE CARD

(continued from page 58)

consequence (force, in our example), is treated in the same manner: The *address* of the match is stored in the *rule_element_type *value* pointed to by the *rule_type *then* side when the appropriate membership element name, PL, is matched.

These steps are repeated for every

rule in the rules file; refer again to the first three rules in Figure 3.

To complete the alterations, other changes included placing the two anchor pointers *System_Output* and *System_Inputs* as global pointers along with the existing *Rule_Base*, adding macro definitions for max and min for cross-compiling onto MS-DOS plat-

forms, and adding the *#include* for the function *strcmp()*; see Example 1. We also included the necessary function to accept two inputs from the command line as arguments for the initial condition *get_system_inputs()* and a function *put_system_outputs()* to examine the exit status of a single inference pass on the input data.

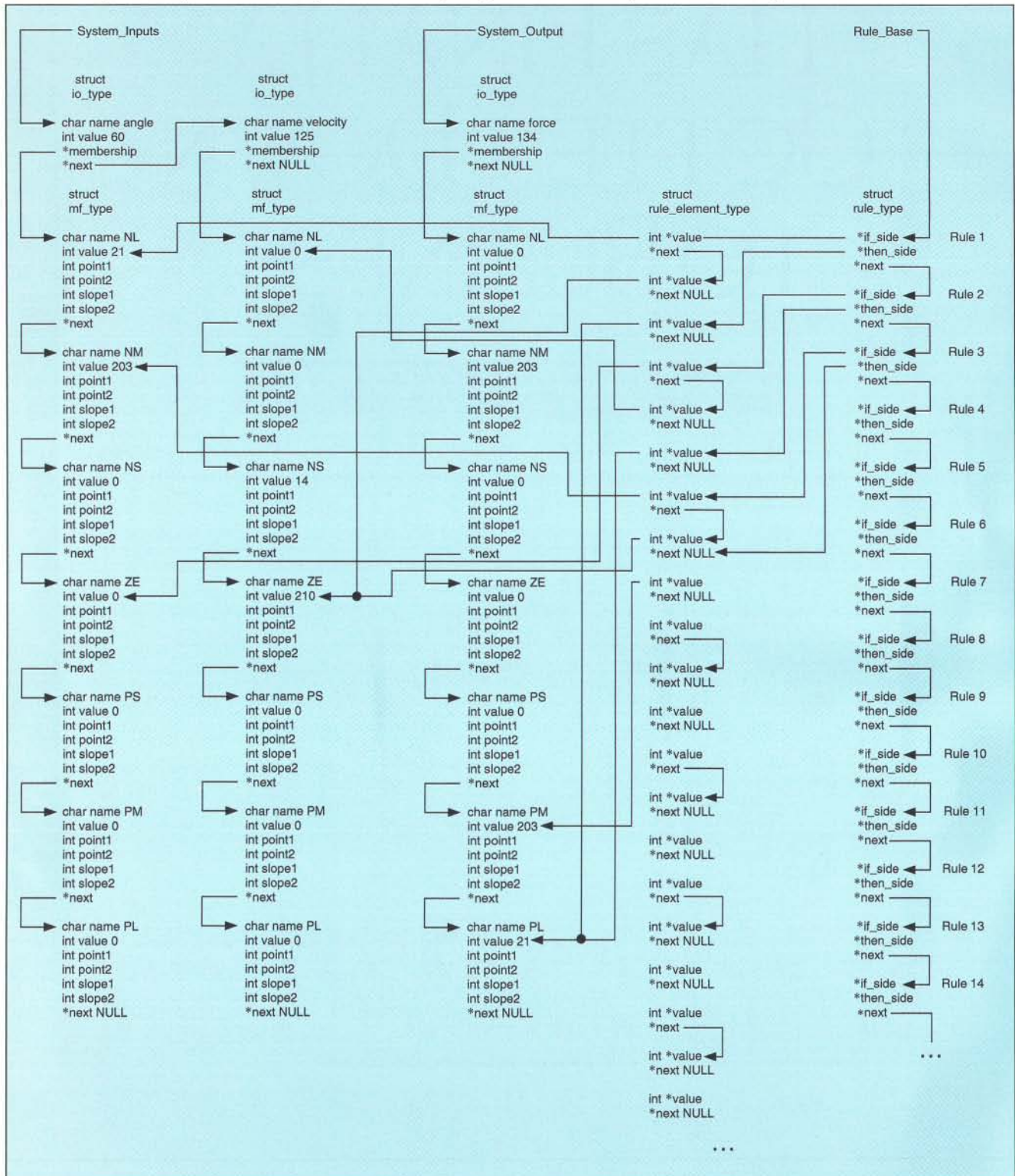
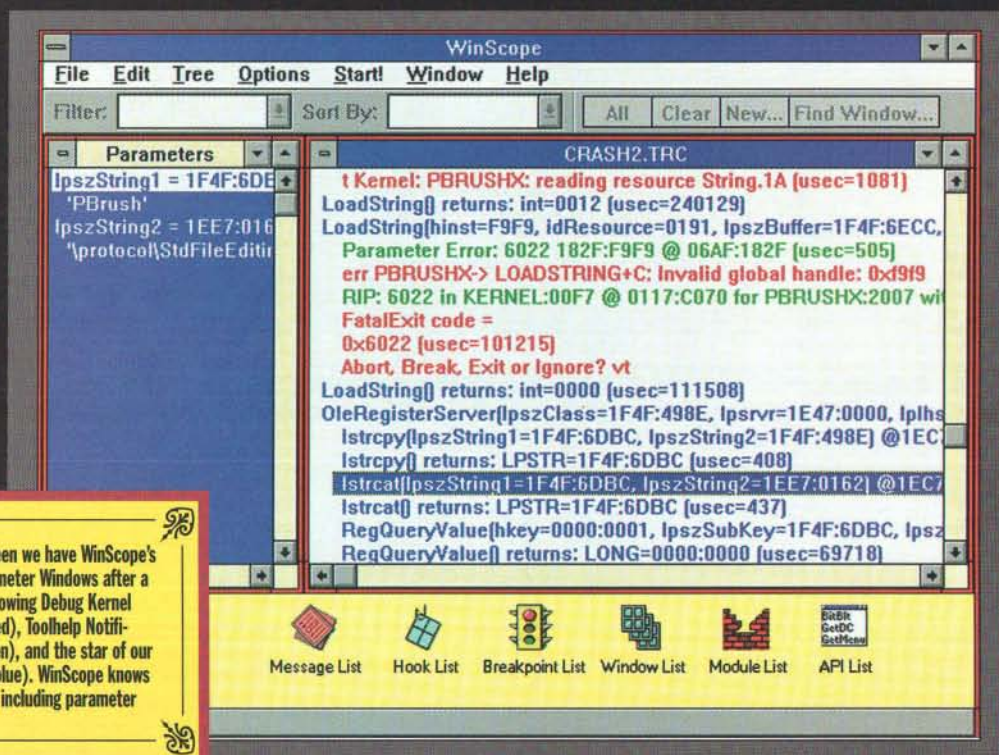


Figure 3: Relationship of data structures and their pointers.

**1993
AWARD
WINNER**

WinScope™

**NATIONALLY
ACCLAIMED
WINDOWS API
DEBUGGER**



On the big screen we have WinScope's Trace and Parameter Windows after a crash. We're showing Debug Kernel Messages (in red), Toolhelp Notifications (in green), and the star of our show, APIs (in blue). WinScope knows all and tells all, including parameter names!

Spotlight problems at the Windows API level. You've had traditional register and source level debuggers for a long time. You've found these debuggers great as far as they go, but their limitations in the event-driven Windows environment have added more frustration (and hours) to your programming life than you care to think about. Traditional debuggers just don't give you the high-level perspective you need to figure out what's going wrong in the Windows environment. And because they're intrusive, they often change or destroy the very clues you need to find the source of a problem.

If you've been dreaming of a debugger that can watch what's going on when your program runs, without interfering... a debugger that can lead you to the source of problems at the API level, showing you a history of pertinent Windows events (API calls with Parameters, Messages, Hooks, Toolhelp notifications, and Debug kernel messages)... your dream just came true.

Discover the secrets of Windows. WinScope is not just an API level debugger. It's also a Windows discovery tool that takes you behind the scenes of Windows itself, to show you how it works, and how other Windows applications do things. When you use WinScope, there are **No More Secrets™**.

**NEW LOW PRICE,
ONLY \$149.
CALL 800/722-7006
TODAY ...
(60-DAY MONEY BACK
GUARANTEE)**



WinScope™



Are you avoiding version control?



Version control isn't just for teams!

If you've ever accidentally deleted source crucial to a project, or introduced a bug into working code, you **know** you need a version control system. Until now VCS's have been expensive and notoriously difficult to maintain. Introducing *Source Control*: finally, individual programmers can track incremental code changes, just like large companies do for their most important projects!

Think of it as object-oriented version control.

Source Control's "project orientation" will take full advantage of your object-oriented code by allowing you to share code among active projects.

Developing for multiple platforms?

Source Control is available now for **DOS, Windows, NT, Macintosh, OS/2 and UNIX**. All of these versions are transparent across a network so you can work with one integrated code base rather than managing time consuming, unnecessarily redundant systems.

If you are part of a team...

Source Control grows easily to meet the needs of team development. A single user can get started for less than \$300 (list price) and licenses are available in increments of five, ten and more. You can economically install *Source Control* on your server and get instant, project-wide status reports. Find out the who, what, where and when of each component as you speed your team to that final build deadline.

Upgrade from PVCS!

Make a quick transition from your current configuration management system with *Source Control*'s PVCS conversion utility. Add Powerline's *make* utility, *Source Make*, and automatically update files without time consuming manual check-in/check-out procedures. Start on your way to a fully integrated configuration management system!

Call for more information!

Powerline Software is dedicated to providing a suite of compiler and platform independent developer's tools, including *Source Print+*, for source code management and *Source View*, the run-time debugger for C and C++.

SOURCE CONTROL

SOURCE MAKE

1-800-257-5773

POWERLINE
SOFTWARE INCORPORATED

1306 Western Avenue, Suite 203, Seattle, WA 98101
206-623-9204 voice 206-467-6561 fax

CIRCLE NO. 898 ON READER SERVICE CARD

FUZZY LOGIC

rule 1: IF (angle is NL) AND (velocity is ZE)
THEN (force is PL)

Figure 4: Sample rule used to develop Figure 3.

```
fuzz 60 125
angle: Value=60
NL: Value 21 Left 0 Right 63
NM: Value 203 Left 31 Right 95
NS: Value 0 Left 63 Right 127
ZE: Value 0 Left 95 Right 159
PS: Value 0 Left 127 Right 191
PM: Value 0 Left 159 Right 223
PL: Value 0 Left 191 Right 255
velocity: Value=125
NL: Value 0 Left 0 Right 64
NM: Value 0 Left 31 Right 95
NS: Value 14 Left 63 Right 127
ZE: Value 210 Left 95 Right 159
PS: Value 0 Left 127 Right 191
PM: Value 0 Left 159 Right 223
PL: Value 0 Left 191 Right 255
force: Value=134
NL: Value 0 Left 0 Right 63
NM: Value 203 Left 31 Right 95
NS: Value 0 Left 63 Right 127
ZE: Value 0 Left 95 Right 159
PS: Value 0 Left 127 Right 191
PM: Value 203 Left 159 Right 223
PL: Value 21 Left 191 Right 255
Rule #1: 21 210 21
Rule #2: 0 0 21
Rule #3: 203 210 203
Rule #4: 0 0 203
Rule #5: 0 210 0
Rule #6: 0 14 0
Rule #7: 0 0 0
Rule #8: 0 210 0
Rule #9: 0 0 0
Rule #10: 0 210 0
Rule #11: 0 14 0
Rule #12: 0 0 203
Rule #13: 203 210 203
Rule #14: 0 0 0
Rule #15: 0 210 0
```

Figure 5: Output generated with a scaled angle of 60 and scaled velocity of 125.

```
#include <string.h>
#define max(a,b) (a<b ? b : a)
#define min(a,b) (a>b ? b : a)
struct io_type *System_Inputs;
struct io_type *System_Output;
```

Example 1: Adding the #include, global pointers, and macros.

```
(a) if(sum_of_areas==0)
{ printf("Sum of Areas = 0, will cause div error\n");
printf("Sum of Products= %d\n",sum_of_products);
so->value=0;
return;
}

(b) int nomatch=0;
for(tp=rule->then_side;tp!=NULL;tp=tp->next)
{ *(tp->value)=max(strength,*(tp->value));
if(strength>0)nomatch=1;
}
if(nomatch==0)printf("NO MATCHING RULES FOUND!\n");
```

Example 2: (a) Code added to the original function defuzzification(); (b) code added to rule_evaluation().

(continued from page 60)

After using the code with various inputs, we needed to add error traps because we were getting core dumps with certain input. These were caused by division by zero when there were no rules in the set to cover the condition. Consequently, we added the code in Example 2(a) to the original function *defuzzification()*. We also added Example 2(b) to *rule_evaluation()*.

After using the code with various inputs, we needed to add error traps because we were getting core dumps with certain input

To further illustrate how you can use the program, assume the scaled angle of 60 and a scaled velocity of 125 as in Figure 5. The line *force: Value=134* reflects the defuzzified and scaled single-valued output for the two inputs. It would be instructive to interface this program to a graphics output device where a loop could be created and the inverted pendulum balanced. Alternatively, a batch file or shell script could feed new inputs and use the output to generate the two new inputs, storing intermediate data in a file. Or, you might try graphing the trapezoidal output areas made on each iteration.

DDJ

(Listing begins on page 101.)

To vote for your favorite article, circle inquiry no. 6.

Introducing EasyCASE[®] 4.1 for Windows:™ Workgroup Edition.

Designing complex systems with teams can spawn its own set of challenges. But with the new EasyCASE Workgroup Edition, you can afford to give your *entire department* the tools it needs to build easily maintainable systems. Based on EasyCASE 4.0 for Windows—*Data Based Advisor* magazine's Best CASE/Design Tool for 1993—EasyCASE Workgroup Edition offers you concurrent, multi-user access to shared projects on your network server.



Breed sound design and security, team-wide.

You'll get locking at the chart, data-dictionary record level and project-wide level; user passwords, permissions; read-only and lockout modes; plus access control.

Develop client/server database systems with the schema generator.

You won't pay extra for our schema generator. Use it for xBASE (dBASE III/IV, FoxPro, Clipper, Paradox), plus a wide range of SQL databases (Oracle, Ingres, Informix, XDB, Progress, DB2, Sybase, SQL Server, Watcom SQL, SQLBase, etc.) You can even reverse engineer xBASE. EasyCASE is also helpful in client/server application development.

Arm your team with the most methodologies.

EasyCASE gives you the *largest* selection of structured analysis, design and data modeling methodologies available in *one* tool:

Yourdon/DeMarco, Gane & Sarson, SSADM, Ward-Mellor, Hatley, Yourdon/Constantine, Martin Information Engineering, Shlaer-Mellor, Chen, IDEF1X, and Merise. Plus a powerful data dictionary repository with extensive reporting capabilities.

EasyCASE

Building
Your Blueprint
for Better System
Design.

Only EasyCASE
gives all this *power*
to all your people.

Easy to master in Microsoft[®] Windows.™

Enjoy a consistent graphical user interface; object-oriented editing; easy chart linking; rules; integrated data dictionary; reports and analysis. First-time users become productive quickly with a tutorial, on-line help, methodology guide and sample projects. Also available: EasyCASE 4.0 for MS-DOS[®] (single user version only).

The highest quality systems for the lowest cost per seat.

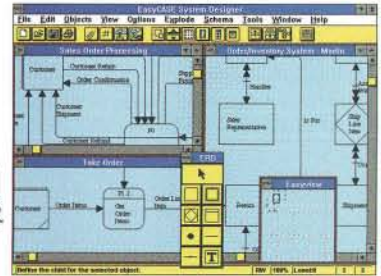
EasyCASE is the undisputed price/performance leader.



No other product in its category gives you so much value, power, flexibility and return on investment as EasyCASE Workgroup Edition. Over 12,000 users worldwide rely on EasyCASE to assure higher quality real-time and information systems. Fact is, you can't afford not to have it.

Look for these features coming soon.

Enhanced client/server support, reverse engineering of SQL, links to PowerBuilder[™] and SQLWindows[™], object-oriented methods, version control, 32-bit version and OLE. Call for availability.



Evergreen
CASE
Tools

"EasyCASE's exceptional charting tools, and numerous supported process and data methodologies put the product well above the pack for products in this price category...The combination of useful CASE tool functionality and low price makes EasyCASE 4.0 a winner!"—PC Week, July '93

For price and product information, call:

1-800-929-5194

Outside the U.S., call (206) 881-5149
8522 154th Ave. NE, Redmond, WA 98052
FAX: (206) 883-7676

Before your team hatches another system design,
make sure it's equipped to avoid problems, not multiply them.

CIRCLE NO. 489 ON READER SERVICE CARD

Digital I/O with the PC

Putting the parallel port to work

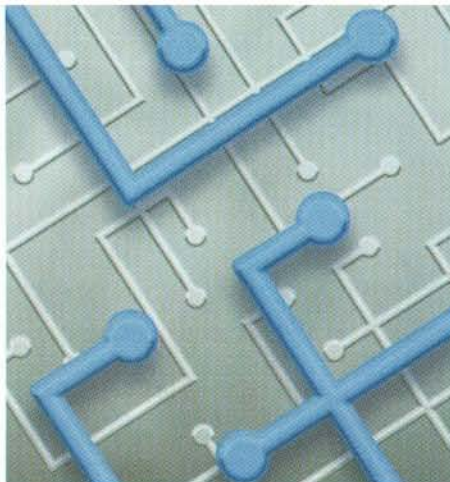
Brian Hook and Dennis Shuman

Data-acquisition and analysis is often performed with dedicated, proprietary, and expensive laboratory instruments. However, the PC's open architecture makes it a cost-effective alternative for many data-acquisition and analysis projects. One particular project we developed at the Acoustic/Electronic Insect Detection Laboratory at the United States Department of Agriculture, Agricultural Research Service facility in Gainesville, Florida required just such a system. The system is an integrated hardware/software setup that allows for digital input and output in a low-end PC configuration. This article describes what we learned while implementing digital I/O via the PC's parallel port.

The system, known as "EGPIC" (Electronic Grain Probe Insect Counter), checks for insects in stored-grain bins and elevators. It does this by electronically sensing insects that crawl into specially designed probes placed at a number of locations in the grain mass. The hardware side of the EGPIC is responsible for detecting an insect and generating the appropriate signal for some

Brian is a programmer at the USDA developing data-acquisition and analysis software. He can be reached on the Internet at bwb@cis.ufl.edu or on CompuServe at 72144,3662. Dennis, a research scientist at the USDA, develops electronic/acoustic systems to detect insect pests in agricultural commodities. He can be reached at USDA, ARS, 1700 SW 23rd Dr., Gainesville, FL 32608.

digital-input computer interface. The software is responsible for reading, analyzing, displaying, and storing the collected data. We selected the PC as the host system for the software because of its wide, low-cost availability and profusion of development tools.



Given these design criteria and the cost and compatibility constraints, the final specification sheet for our digital input and output (DIO) interface was as follows:

- Simple interface to a PC.
- Multiple digital input lines.
- Interrupt-on-input capability.
- At least one digital output, preferably more than one.
- Compatibility and availability across a wide range of platforms, including ISA, EISA, and MCA buses.
- Relatively low cost.

Digital I/O Options

The PC architecture has a wide variety of input and output techniques available to it, from specialized DIO boards to the relatively crude game port. Each has some advantages and disadvantages for this type of system.

Specialized DIO boards are available for the PC, typically as 8- or 16-bit ISA boards with 48 I/O lines, configurable

to generate interrupts on one of several different IRQs. While nearly ideal feature-wise, these boards are relatively expensive—from \$40 or \$50 to more than \$1000. Even with the inexpensive boards, this cost becomes significant in high volumes. In applications using from 9 to 96 probes, EGPIC has been configured for use with these DIO boards. However, when only one to eight probes are needed, the DIO board is unnecessary. Also, since these boards require a free bus slot, some systems, such as laptops (a likely target platform), would be excluded from using EGPIC.

The PC's standard RS-232 serial port is suitable for this type of application, but the external EGPIC hardware would require an extra translation layer to generate RS-232-compatible bit streams from the eight digital inputs. This method is suitable for very large systems requiring hundreds or thousands of probes, but is unnecessarily complex for smaller-scale systems.

Among other deficiencies, neither the keyboard input nor the game port offer output capabilities, ruling out their consideration as suitable I/O interfaces for EGPIC.

This leaves the PC's printer parallel port. Like a dedicated DIO card, it offers digital output lines and interrupt-on-input capability (using either IRQ 5 or 7). Unlike DIO cards, it's available for all PC platforms and is relatively inexpensive. The parallel port's only shortcomings are that not all implementations have input capability and the port may already be in use by another device, likely a printer. However, many systems have two parallel ports; if not, a second parallel port is an inexpensive addition. And as for the lack of input capability, after some software tricks, a 100 percent compatible PC parallel port can, in fact, be used for up to 8 bits of digital input.

Programming the Parallel Port

To illustrate the programming techniques discussed in this article, I've writ-

Considering desperate measures?

No need, when Codewright has what you are looking for.

Today's technology in a Programmer's Editor

Splitting windows and column blocks were "hot technology" in 1985. If you need today's technology for today's platform, you need Codewright. With Codewright, you get the power of Projects and Workspaces, Color Highlighting, and a Class Browser without confining yourself to an IDE.

No IDE confinements

Codewright leaves you free to use a variety of tools. It provides Multi-language support, Version Control support, Multi-file operations, and, most importantly a very powerful editor. Configure or extend it to fit your every need.

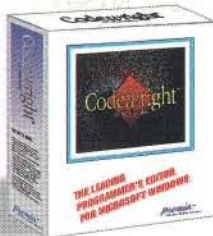


"...a programmer's dream"

Richard Hale Shaw
PC Magazine, Oct. 27th 1993

Codewright empowers you

Like the finest editors of the past, Codewright empowers you with configurability and extensibility. In fact, our simple DLL interface makes Codewright the most extensible editor available. Codewright also empowers you with a Hex editing mode, Selective Display mode, side-by-side File Comparisons and Difference Merging.

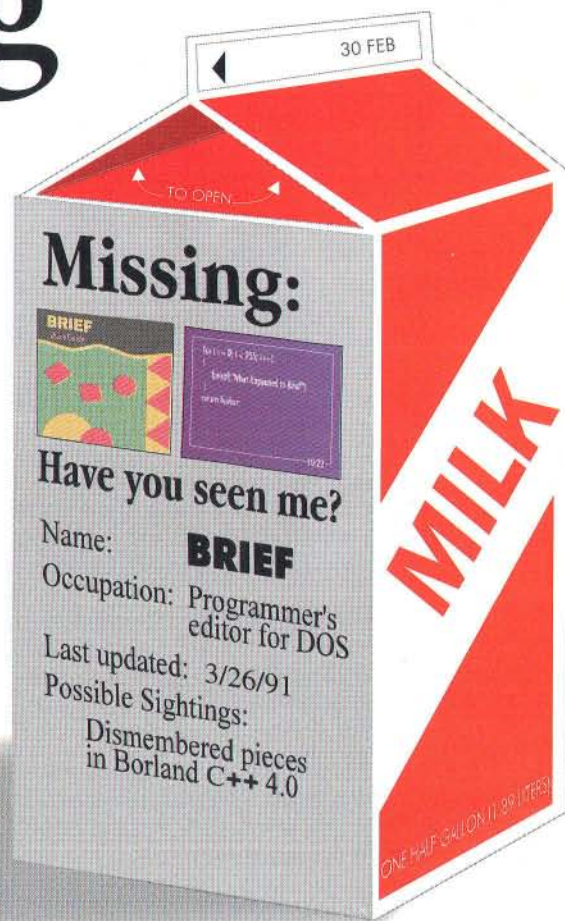


Premia
Premium Quality Software

SD '94
Booth #1111

Premia Corporation
1075 NW Murray Blvd., Suite 268
Portland, Oregon 97229 USA
Fax: 1-503-641-6001 Phone: 1-503-641-6000

FEATURE	Codewright	Borland IDE 4.0	Brief
Maximum # of files	Unlimited	128	Unlimited
Keystroke record and playback	Yes	No	Yes
Multi-buffer search and replace	Yes	No	macro
Language templates	Yes	No	Yes
Multi-compiler/language support	Yes	No	Yes
Smart indenting	Yes	No	Yes
Selective display mode	Yes	No	No
Hex display/edit mode	Yes	No	No
Project support	Yes	Yes	No
Version control integration	Yes	No	Yes
File differencing and merging	Yes	No	No
Drag and drop text editing	Yes	No	No



Codewright makes switching painless

In addition to CUA, Codewright can emulate other editors, including BRIEF, Epsilon, vi and WordStar. It can even run most BRIEF macros. Use it with your help files. It even integrates with your IDE so you won't feel like you're missing anything.

Give us a call

Try Codewright now, at our risk, and you'll see you don't have to get desperate to find what you need. It's right here in Codewright.

Codewright™
for WINDOWS
and WINDOWS NT **V3.0**

Editor of Choice for Professional Programmers

Your choice!
Windows or Win32 version
Same price.

Single-user license:

\$249

Volume discounts and site licenses available.
Call for information.

CALL NOW!
1-800-547-9902

Premia is a registered trademark of Premia Corporation. Codewright is a trademark of Premia Corporation. Brief is a registered trademark of Borland International.

CIRCLE NO. 826 ON READER SERVICE CARD

MODEL IT.

VISIONNAIRE™

© 1993 AT&T

Creating software programs with just the right fit for your customer's needs is what VISIONNAIRE software is all about. A high-quality specification development tool, VISIONNAIRE's interactive multimedia environment enables animated modeling of systems, products and services. The result is higher quality functional specs, enabling more effective test planning and implementation. That increases your productivity by cutting costs and decreasing lengthy development intervals. And VISIONNAIRE can also reduce the sales cycle, since it's very skillful at getting across the features and benefits of products difficult to demonstrate.

For software that shows what you're thinking, call 1 800 462-8146 or fax 1 908 580-6355.

SOFTWARE SOLUTIONS GROUP

Innovation you can depend on.

CIRCLE NO. 801 ON READER SERVICE CARD



DIO

(continued from page 64)

ten the Parallel Port Digital Input Output (PPDIO) package, a rudimentary set of C functions that allow for reading and writing to the parallel port and installing an interrupt-service routine (ISR) to handle incoming data on the parallel port. Listing One (page 103) is PPDIO.H; Listing Two (page 103) is PPDIO.C.

The parallel port is programmed via three separate I/O registers: the input-only data register, the output-only status register, and the input/output control register. The addresses of these register ports differ depending on the machine, but they are usually offset from 0x378, 0x278, or 0x3BC. The base address for a particular LPT port is stored in the BIOS data area. The *PPDIO_GetLptAddress()* routine shows how to retrieve this information.

The data register (see Figure 1), located at the parallel port's base address, takes a standard bit mask that indicates which pins should be sent high and low. Sending information out the parallel port is accomplished with a simple OUT instruction. *PPDIO_SendByte()* handles this. The parallel port transmits this byte until told to transmit a different one, making digital output a trivial task. Note that while we can theoretically read the data register with an IN instruction, the byte read won't be incoming data—it will be the most recent data transmitted.

The data register can't be used for input, so we must use both the status and control registers; see Figures 2 and 3 for their respective layouts. Reading the status register is very straightforward, but keep in mind that the logic of pin 11 is inverted.

The control register is nominally an

Bit	Pin/Function	Logic
0	0	1=TRUE
1	1	1=TRUE
2	2	1=TRUE
3	3	1=TRUE
4	4	1=TRUE
5	5	1=TRUE
6	6	1=TRUE
7	7	1=TRUE

Figure 1: Data register (base address+0).

Bit	Pin/Function	Logic
0	(reserved)	—
1	(reserved)	—
2	(reserved)	—
3	15	1=TRUE
4	13	1=TRUE
5	12	1=TRUE
6	10	1=generate interrupt
7	11	0=TRUE

Figure 2: Status register (base address+1).

NEW! ROM-DOS™ 6

A fully compatible, ROMable DOS for embedded systems

Gives you MORE...

- ▶ **More Embedded Tools:** ROMable EXE's, Automated BUILD Configuration, royalty free miniBIOS, optional data compression and PCMCIA support.
- ▶ **More Flexibility:** Configure ROM-DOS the way you want. Device drivers in source let you support non-standard features or calls.
- ▶ **More Support:** Our technical staff assists you until your system is up and running, even on non-DOS issues.



and LESS...

- ◀ **Less ROM and RAM Used:** ROM-DOS is half the size of MS-DOS®.
- ◀ **Less Money per Copy:** Save up to 80% compared to MS-DOS.
- ◀ **Less Risk:** No-nonsense 90 day, 100% Money Back Guarantee on ROM-DOS Software Developer's Kit.

FREE Bootable Demo Disk! Call 1-800-221-6630

307 N. OLYMPIC, SUITE 201 • ARLINGTON, WA 98223 USA • (206) 435-8086 • FAX: (206) 435-0253

Datalight®

Developer Tested Only. Novell makes no warranties with respect to this product. • MS-DOS and Microsoft are registered trademarks of Microsoft Corporation.



Make it fly. Paradigm DEBUG



Looking to soar with ease through the toughest debugging scenarios? Whether running stand-alone with your target system, or as a front-end to your favorite in-

circuit emulator, we have a version of Paradigm DEBUG that has what it takes to help you reach into the design stratosphere.

Paradigm DEBUG teamed with Paradigm LOCATE – you've got it all. Intel 80C186Ex or NEC V-Series microprocessors. Borland C++ and Microsoft C/C++ compilers. Stand-alone or hardware-assisted debugging. Unlimited toll-free technical support. Satisfaction guaranteed.

No one else even comes close to providing the total solution that Paradigm has delivered for more than two years running.

So whether you're currently stuck on the ground, or getting ready to launch a new application, give the embedded system design experts a call. We'll rocket you the complete details on Paradigm DEBUG and see about getting you your own set of wings.

PARADIGM

Proven Solutions for Embedded C/C++ Developers

1-800-537-5043

Paradigm Systems
3301 Country Club Road, Suite 2214
Endwell, NY 13760
(607) 748-5966 • FAX: (607) 748-5968

All trademarks are property of their respective holders.



Bit	Pin/Function	Logic
0	1	0=TRUE
1	14	0=TRUE
2	16	1=TRUE
3	17	0=TRUE
4	IRQ enable	1=enabled
5	(reserved)	—
6	(reserved)	—
7	(reserved)	—

Figure 3: Control register (base address+2).

(continued from page 66)

output-only register, but by taking advantage of the four output lines driven with open-collector drivers, we can force the control register into giving us input. If we produce a high TTL logic level at the control register's corresponding pins, we can drive the pins low via incoming signals. Thus, by setting the appropriate bits of the control register, the pins can be used as input. This is handled transparently when `PPDIO_InstallISR()` is called.

Reading the control and status registers is accomplished by an IN instruction at the port's base address and relevant offset. The routines `PPDIO_ReadStatusRaw()` and `PPDIO_ReadControlRaw()` illustrate how to accom-

plish this. Because several of the input lines have negative active logic placed upon them by the parallel port, helper functions that translate negative logic would be useful. The routines `PPDIO_`

Like a dedicated DIO card, the PC's parallel port offers digital output lines and interrupt-on-input capability

`ReadStatusCooked()` and `PPDIO_ReadControlCooked()` provide this functionality, along with converting reserved and unused bits to 0.

Interrupt-driven Communications

Now that input and output have been addressed, all that's left is making the communications interrupt-driven. The

parallel port's input lines could be polled; however, this would be cumbersome, time consuming, and error-prone. Having an interrupt generated whenever a digital line is sent high is a far more elegant means of input detection.

Assuming the target system supports it, interrupt-driven input on the parallel port is actually not very complicated to achieve. First, the control register must have its interrupt-enable bit set. Next, an ISR must be installed in the DOS interrupt vector table for the appropriate IRQ. Finally, the port's IRQ must be unmasked from the 8259 Programmable Interrupt Controller's interrupt-enable register. All of this is demonstrated in `PPDIO_InstallISR()`.

While programming in an interrupt-driven manner is theoretically simple, hardware support can be shaky. The printer port has traditionally utilized IRQ 7, but IRQ 5 isn't uncommon either. Even worse, some machines either have the parallel-port interrupt disabled altogether or have an alternate device (such as a network or sound card) using its IRQ. To compound matters, there's no easy way to detect which IRQ a given base address or LPT port corresponds to—this must either be known by the user or determined empirically by the program.

Real-Time Multitasking

for Microsoft C, Borland C, Borland Pascal

Develop Real-Time Multitasking Applications with RTKernel!

RTKernel is a professional, high-performance real-time multitasking kernel. It runs under MS-DOS or in ROM and supports Microsoft C, Borland C++, Borland/Turbo Pascal, and Story Brook Pascal*. RTKernel is a library you can link to your application. It lets you run several functions/procedures as parallel tasks and offers the following advanced features:

- pre-emptive, event/interrupt-driven scheduling
- number of tasks only limited by RAM
- task-switch time of 6 μ s (33-MHz 486)
- up to 64 priorities (changeable at run-time)
- time-slicing can be activated
- timer interrupt rate of 0.1 to 55 ms
- high-resolution interval timer (1 μ s)
- activate or suspend tasks out of interrupt handlers
- programmable interrupt priorities
- inter-task communications using semaphores, mailboxes, and message-passing
- keyboard, hard disk, floppy disk idle times usable by other tasks
- interrupt handlers for keyboard and COM ports included with source code
- supports up to 36 COM ports (DigiBoard and Hostess boards)
- protocols XOn/XOff, DTR/DSR, RTS/CTS
- full support of NS16550 UART chip
- supports math coprocessor and emulator
- fast, inter-network communication using Novell's IPX services
- runs under DOS, LANs, or without OS
- call DOS without re-entrance problems
- resident multi-tasking applications (TSRs)
- runs Windows or DOS Extenders as a task
- supports CodeView and Turbo Debugger
- Kernel Tracer for easy debugging
- ROMable
- full source code available
- no run-time royalties
- free technical support by phone or fax

RTKernel-C 4.0 \$495

RTKernel-Pascal 4.0 \$445

C Source Code: add \$445

Pascal Source Code: add \$375

International orders: add \$30 shipping and handling. MasterCard, Visa, check, bank transfer accepted.

FREE DEMO DISK
Please request Info Kit D

On Time
MARKETING

Professional Programming Tools

In North America, contact:
LEL Computer Systems
20 Canterbury Ct. • Setauket, NY 11733
USA • CompuServe 73313.3177
Phone (516) 473-8119 • Fax 331-0706

Outside North America, contact:
On Time Marketing
Karolinenstrasse 32 • 20357 Hamburg
GERMANY • CompuServe 100140.633
Phone +49 - 40 - 43 74 72 • Fax 43 51 96

MS-DOS 6.2

Now embedded systems developers can get MS-DOS 6.2 licenses from Annabooks, even in small quantities. And now you can also license Windows 3.1, Windows for Workgroups 3.11, and Windows NT from us!

We also have new lower prices for MS-DOS 3.xx, 4.xx, and 5.0. If you need to build diskless systems, PromKit will let you boot from ROM. And some of these MS-DOS versions even execute out of ROM, if that's what you need.

Call now for more information about OEM Developer's Kits, PromKit, and license procedures and costs. You will be amazed at how easy it is!

Annabooks

11848 Bernardo Plaza Ct., Suite 110
San Diego, CA 92128-2417

1-800-462-1042 619-673-0870 619-673-1432 FAX

We accept Visa,
MasterCard, AmEx,
and company POs

Microsoft and MS-DOS are trademarks of Microsoft Corporation

CIRCLE NO. 394 ON READER SERVICE CARD

CIRCLE NO. 622 ON READER SERVICE CARD

80C196

In-Circuit Emulators

Microsoft Windows 3.1 based user interface

Pull-down Menus

Speed bar (point and click)

On line help

Disassembler,
in-line
assembler
Window

Register
Window

Data
Window

Shadow RAM
shows data
changes in
real-time

Watch
Window
for high level
variables

Source
Window

Call Stack
Window

Inspect
Window

Context sensitive help line

Trace Buffer Window



FEATURES:

- Support for 80C196: KR/Q/T/C/D, JR/Q, NQ/T & more.
- Real-time emulation at maximum chip speeds.
- Use of bond-out chips for accurate emulation.
- Hosted on PC's and workstations.
- High Level support for popular C-compilers.
- Unlimited hardware breakpoints.
- Break in real-time on Internal Access, both on data value and address.
- Trace board up to 512K deep, 104 bits wide, with 40 bit timestamp. Triggering and filtering with full instruction queue decoding.
- Memory contents shown during real-time emulation (Shadow RAM).
- Code Coverage and Program Performance Analysis.
- CCB's controlled from user interface. (Wait states, timing mode, bus width and more.)

Also supported:

8051 68HC11 68HC16 683xx

To learn more, please call (408) 866-1820 for a FREE Demo Disk. For more information via your Fax, call our 24-hour Information Center at (408) 378-2912.

NOHAU
CORPORATION

51 E. Campbell Avenue
Campbell, CA 95008-2053
Fax: (408) 378-7869
Tel. (408) 866-1820

Argentina 1 312-1079, Australia (02) 654 1873, Austria 0277 20-0, Benelux (01858) 16133, Brazil (011)-458-8755, Canada 1-514-689-5889, Czechoslovakia 0202-2683, Denmark 43 44 60 10, Finland 90-4526-21, France (1) 69 41 28 01, Germany 49-7043-40247, Great Britain 0962-733140, Greece 1-924 20 72, India (0212) 422164, Israel (03) 491202, Italy (011) 437 15 51, Korea (02) 784-7841, New Zealand 09-3092464, Norway 22-67 40 20, Portugal 01-80 95 18, Romania 961-30078, Singapore 749-0870, S.Africa (021) 23-4943, Spain (93) 276 22 69, Sweden 040-92 24 25, Switzerland 01-740 41 05, Taiwan 02 7640215, Thailand (02) 281-9596.

Only
\$395**Fuzzy Logic
Designer™**The tool to use for applied fuzzy logic,
not hazy theories

Now with neural rule weights

- Generate portable C source, unique to your design
- Includes detailed documentation and tutorial explaining fuzzy logic theory
- Use any debugger, emulator with your application or embedded system
- Provides graphical simulation capability and GUI using Microsoft Windows(™)
- No royalty or license fees for generated output
- Only \$395.00, Call for information today.
- Additional OEM and DLL capability available

**Byte Dynamics, Inc.**14808 E. Olympic Ave.
Spokane, Wa. 99216

Call: (800) 233-2983

Fax: (509) 926-6130

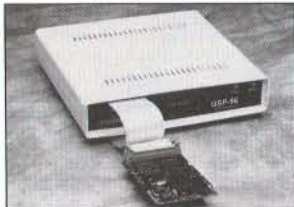
* Custom software development available *

CIRCLE NO. 824 ON READER SERVICE CARD

In-Circuit Emulators

Development tools for the most demanding

8051
80196
80186
HPC +
Z8
8085
DSP's



- Unparalleled features
- HLL debugger with locals support
- External unit, no plug-in cards !!!
- High speed download (64k in 12 sec)
- Banking support for > 64KB operation

see the difference - free 2 week trial

SIGNUM SYSTEMSMountain View, CA / Thousand Oaks, CA.
(415)903-2220 (805) 371-4608

CIRCLE NO. 829 ON READER SERVICE CARD

**ROM-IT
EPROM EMULATION SYSTEM****The Most Flexible
EPROM Emulator
You Can Get Today**

- Emulates up to eight 4-Megabit EPROMs through one standard serial port.
- Downloads 2-Megabit programs in less than 23 seconds
- Examine and modify individual bytes or blocks.
- Accepts Intel Hex, Motorola S-Record and Binary files.
- Software available for IBM PC and compatibles.
- Base 27256 EPROM System \$395.00. Other configurations available.

**Incredible Technologies, Inc.**

Visa, Mastercard and American Express Accepted

Order Now - It's Easy
Call (708)870-7027 Or Fax (708)870-0120
For More Information

CIRCLE NO. 760 ON READER SERVICE CARD

(continued from page 68)

To solve this problem, EGPIC uses a simple call-and-acknowledgment method of IRQ determination. This involves installing ISRs at IRQs 5 and 7, "calling" the EGPIC hardware (which acknowledges the call by generating an interrupt), then seeing which ISR is called.

*By taking advantage
of the four output
lines driven with
open-collector
drivers, we can
force the control
register into giving
us input*

If no ISR is called, either another IRQ is in use (doubtful, since the PC industry has fairly well standardized on IRQs 5 and 7 for the parallel port), or no IRQs are being used for the parallel port. It's simpler to have the user input which IRQ to use, but this demands a higher level of user knowledge than the application may reasonably assume. An interesting secondary use of this call-and-response procedure is for hardware testing—if a probe is known to be installed and fails to generate a response when requested, then that probe must be malfunctioning.

Interrupts are generated via pin 10, normally a printer's ACK line. A high-line level sent to pin 10 results in an interrupt being generated, assuming that all other relevant setup has been done.

During the development of EGPIC we found that both cable length (from the probe to the computer) and interrupt latency played a role in determining whether a signal actually existed at the inputs when the ISR was called. With long cable lengths and a fast computer, it was possible for some inputs not to be updated by the time the ISR was executed. Conversely, with a slow computer it was possible for the signal to have come and gone (depending on the length of the generated input) by the time the ISR was called. These timing problems can be compensated for in hardware, but not knowing of their existence can lead to some rather irksome bugs.

Potential Problems

Two particularly bothersome problems came to light while developing the EGPIC system.

First, if the interrupt lines were allowed to float while the system was collecting data, the PC would likely lock up. This is because a floating line will often fluctuate between TTL TRUE and FALSE, causing thousands of interrupts to be generated every second, freezing up the system. Something as innocuous as accidentally pulling a cable loose or turning off the input hardware may cause a system lockup.

The second problem is that not all PC parallel ports are identical. Some parallel ports deviate considerably from the original PC's design, rendering this type of specialized input and output—which assumes 100 percent hardware compatibility—impossible. Unfortunately, only trial and error will determine which systems are nonstandard.

Conclusion

Listing Three (page 103) is DIO.C, a program that implements simple interrupt-driven DIO with the parallel port. By itself, DIO.C is fairly useless—consider it more of a basic framework to draw upon than a real program. Any program that would use these routines would require some amount of custom hardware and software design.

At first glance, the antiquated design of the PC parallel port seems highly limited, but it can quite easily be configured as an inexpensive digital input/output interface. The hardware required is minimal: a one-shot circuit per channel and a single interrupt line for the logical OR of all the channels. The software, as demonstrated in this article, is quite simple and easily customized for specific applications. In our case, the parallel port satisfied all of our requirements superbly, enabling the EGPIC project to be both completed in a timely and cost-effective manner and distributed across a wide range of systems.

Acknowledgments

We are grateful to Hok Chia and Sergey Kruss (University of Florida) for electronic technical assistance.

References

EGgebrecht, Lew. *Interfacing to the IBM Personal Computer*, Second Edition. Carmel, IN: SAMS, 1992.

DDJ

(Listings begin on page 103.)

To vote for your favorite article, circle inquiry no. 7.

ARRESTED DEVELOPMENT?

Come see us
at the Embedded
Systems Conference
Booth #625

Let Diab Data's D-TECTIVE Read Your Bugs Their Rights!

Are your efforts to get production-quality software out the door being dogged by suspicious, recurring, and persistent bugs? Put Diab Data's D-TECTIVE on the beat!

D-TECTIVE is the latest word in GUI, multi-tasking, multi-target, remote, source-level debuggers for Motorola's 680x0 and CPU32(+) family of embedded controllers and microprocessors, and it runs on the most popular platforms, in-circuit emulator tools, and real-time operating systems.

You've already heard about the legendary performance of Diab Data's D-CC/68K—the fastest, most reliable compilers for Motorola's 680x0 family of processors. Now, the industry's recognized leader in high-performance globally optimizing compilers has added proven debugging

technology to its D-CC/68K software development tool arsenal.

Protect your code's performance and quality with D-CC/68K and D-TECTIVE—the smartest one-two punch available anywhere to get your Motorola-based embedded project to market fast and guarantee that it'll be a star performer.

Stop bugs and quality glitches from arresting your development! Let D-TECTIVE read 'em their rights! Call Diab Data now to learn more about the most arresting software tools in the industry and our FREE evaluation program.

DIAB DATA
A Bull Company

U.S.: Telephone: (415) 571-1700 • Fax: (415) 571-9068
Europe: Telephone: +46-8 622-4422 • Fax: +46-8 622-4223

EchoNets, E-memes, and Extended Realities

Mobile computing requires new ways of thinking about networks

Scott B. Guthery

The walkie-talkies of computing are personal digital assistants, or PDAs. Apple's Newton, AT&T's EO, Casio's Zoomer, and IBM's Simon all open the possibility of direct, computer-to-computer connections using wireless personal-communication systems. Network vendors will contend that central switches are necessary for communication among a large number of people, and while central switches do add features to network communication, switchless network communication that relays messages from one node to another is also possible. This article will explore possibilities for switchless networks which I will call "EchoNets."

EchoNets

Suppose that every minute or so your PDA broadcasts a message such as, "Curly here. Anybody out there?" Then, suppose I happen to walk by with my PDA turned on. It answers, "Yeah, Moe here. What's up, Curly?" Yours replies, "I've got 15 messages for you," and sends my PDA the messages. "Thanks," mine says, "and here are eight for you." "See ya, Curly," yours says. "Ciao, Moe," says mine, and they go their separate ways.

This is a basic EchoNet message exchange, a form of which is used in existing computer networks, including Usenet, FidoNet, and Relaynet. In fact,

Scott is a scientific advisor at the Schlumberger Laboratory for Computer Science. He can be reached via Internet at guthery@austin.slcs.slb.com.

the news groups in FidoNet [Bush 93] are actually called "echoes," and mail is called "echomail." The difference between the use of the EchoNet-style protocols by PDAs and their use in existing networks is that the nodes in existing networks exchange messages with nodes that are known and relatively fixed over time. In a PDA EchoNet, a node is constantly polling for and talking to strangers.

Obviously, if I'd queued up a message to you in my PDA or if you'd entered a message to me in yours, we



would have communicated without a switch. This is plain-vanilla, peer-to-peer communication. But suppose my sister had written a message to you last night on her PDA. Sometime during the night, her PDA engaged in a message exchange with mine, and I unwittingly carried her message with me when I left for work this morning. When I walked by your PDA, I delivered her message to you. In a sense, my PDA was the network backbone that carried my sister's message to you.

EchoNets are by no means a recent discovery. One of the earliest networks in the Internet, the DARPA Packet Radio Network, used a "flooding" protocol, which is a form of EchoNet. And

the gateways and bridges in modern switched networks are really nothing more than EchoNet nodes with fixed neighbors. So even switched networks may have EchoNet subnets.

However, we've become so accustomed to the features provided by switches that we think communication systems require them. While it will be useful for PDAs to be able to connect to pay-per-message switched networks such as cellular, telephone, satellite, and cable systems, it's important to realize that you are buying the added features of the switch and billing, as much as the raw network capability itself. It's also useful to realize that you can communicate without them.

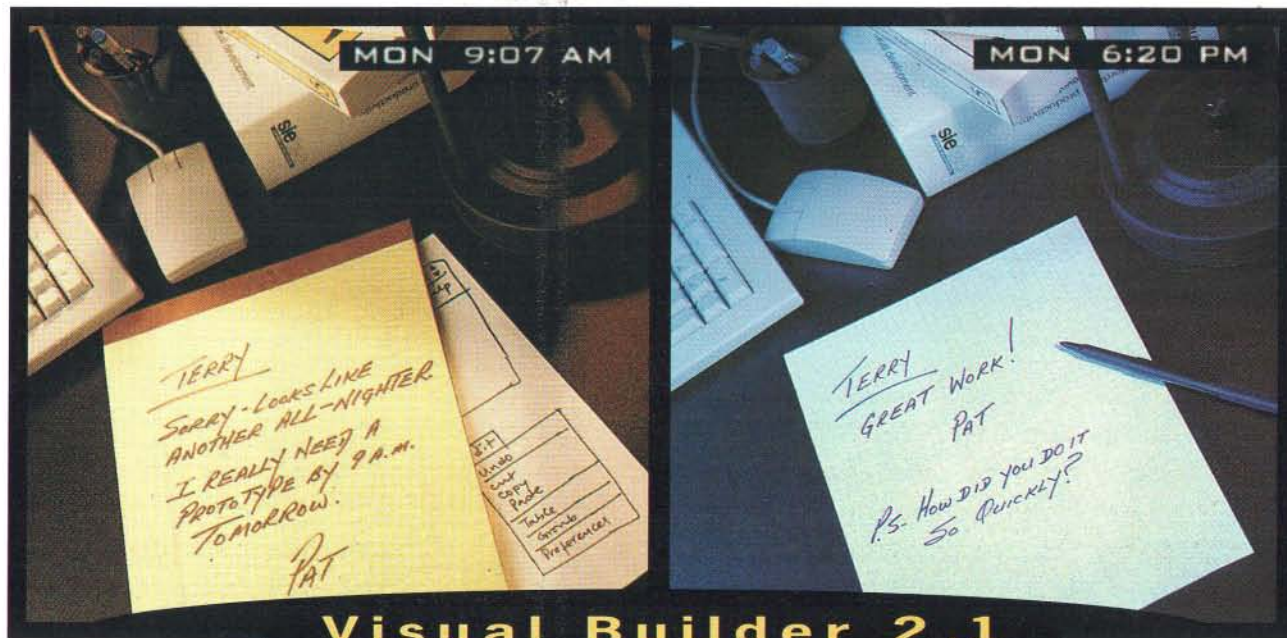
Receiver Addressing vs. Sender Addressing

Of course, one downside of EchoNet messaging is that the mail may not go through. And even if it eventually does go through, you have no idea about nor any control over how long it will take to deliver your message. It's like Usenet or FidoNet e-mail, only worse—much worse.

To get an EchoNet mail message from me to you, we need a chain of PDAs: First, I have to pass near A, then some time later, A has to exchange messages with B, and so forth, until you finally cross paths with Z. While it's helpful that the whole chain of PDAs need never exist in totality at any one point in time, you and I are clearly at the mercy of many chance events and random happenings. What this means is that you can probably get EchoNet mail reliably to people in the crowd you hang out with, but it's unlikely that you can get EchoNet mail to your friend in Tuva. In fact, it could be argued that EchoNets aren't good at all for person-to-person e-mail when you know exactly who you want to send the message to and what their address is.

Where EchoNets beat traditional e-mail (and telephone and surface mail,

Now you can get from A TO B without losing any ZZZZs.



Sleepless nights. A thing of the past for GUI developers?

CONCEPT TO DELIVERY

Visual Builder 2.1 saves you endless hours of tedious coding and fine-tuning of graphical user interfaces. You can visually layout, simulate and modify your GUI to build a solid prototype in just a fraction of the time. And producing a finished GUI application is just as fast when you add behavior to your layout using Visual Builder's simple yet powerful scripts rather than hand-coding C callbacks.

PLATFORM TO PLATFORM

Visual Builder has always made it easy to build GUI based applications on a UNIX® platform and move them to another UNIX platform or to a PC running Microsoft® Windows. Now you can also develop in Microsoft Windows and move your application to UNIX platforms just as easily.

IDT TO UIMS

Visual Builder is available as an Interface Development Tool and as a User Interface Management System, and you can upgrade from IDT to UIMS with only a phone call to activate the UIMS components.

The IDT is a customizable GUI builder with WYSIWYG layout, simulator, automatic C code generator, and interface portability. The UIMS has the same robust features and adds powerful behavior scripting and the ability to integrate additional widgets for both Motif™ and Microsoft Windows.

US TO YOU

Try Visual Builder 2.1 FREE for 30 days. With our carefully designed tutorial you'll be proficient in just a few hours.

- ★ Call us toll-free at 1-800-565-5650
- ★ Fax us at 1-416-496-8524
- ★ Send email to support@sni.ca
- ★ Pull software and documentation off Internet using anonymous ftp at <ftp.uu.net> in the `/vendor/Sietec` directory.

THE SIEMENS HERITAGE

Sietec Open Systems is a division of Siemens in Canada. The Siemens name has always been synonymous with technological innovation - from laying the first trans-Atlantic communications cable to creating robots capable of seeing. With products like Visual Builder, this tradition continues.

Distinct[®] TCP/IP for Windows[™]



Standard

- APIs for Windows Sockets, Berkeley Sockets, ONC RPC/XDR, Telnet, FTP and NetWin.
- Up to 128 concurrent sessions with user definable buffers and timeout values.
- Supports NDIS, ODI and Packet drivers.
- SLIP, CSLIP and PPP with powerful scripting.
- BOOTP and RARP client and server.
- Co-exists with Netware, Lan Manager, Vines, Lantastic and Pathworks at no additional cost.
- Over 200 KB source code sample programs.
- Complete network configuration program with automated network detection technology.

Professional

- Includes all features of the Standard Edition.
- Includes Telnet VT emulator, drag and drop FTP client and server, TFTP, LPR, LPD, transparent printing, Network Monitor, Finger and more.

Visual Edition[™]

- The first and only TCP/IP custom controls for Visual Basic from the winner of the BasicPro readers choice award. Includes custom controls for Windows Sockets, FTP and Telnet.
- Ideal for writing Visual Basic front ends to client/server applications.
- Use as an add on to the other kits or alone.

CIRCLE NO. 838 ON READER SERVICE CARD

408.741.0781

Email: mktg@distinct.com
Fastfacts: 408.867.4742
Fax: 408.741.0795

distinct

```
To: EchoNet Implementers Everywhere
From: Earl of Echoes in Austin
Subject: Improved Short-Hop Protocol
Keywords: EchoNet, Transfer Rules
Send-Date: September 6, 1993
Route: Bubba in Temple, Jenny Jet
      in Dallas
```

Figure 1: An EchoNet header field.

(continued from page 72)

for that matter) is when you don't know who you want to send the message to, or when you don't know how to get in touch with them. In this case, you want to broadcast your message in the hopes that the person or people you're looking for will receive it. Thus, it is the receiver or receivers of the message, rather than the sender, who determines to whom the message is addressed.

Just like a piece of e-mail, an EchoNet message comes with a number of header fields (see Figure 1) which describe it. Header fields help you scan the incoming EchoNet messages quickly to find the ones that are meant for you. You'll probably want to activate some sort of automatic text filter to sift through all the incoming messages and set aside ones that fit you and your profile of interests.

E-memes

It's important to remember that, unlike Usenet newsgroup messages, you're both a potential recipient and a relay point for all of the EchoNet messages you receive. Just because you find a message you think is addressed to you, doesn't mean you shouldn't pass it on. There may be other people who are interested in the message, and you're part of the chain that is going to get it to them.

By default, you should also relay messages that aren't addressed to you along with ones that are. On the other hand, you're free to look over your message traffic and delete any messages you don't want your PDA to pass along. In an analogy to the memes (or thoughts) of human communication, EchoNet messages are kind of like e-memes. E-memes that people like—that they want to tell other people about—are passed on. E-memes that people don't like die off quickly, either by being read and deleted, or by being killed by automatic purge rules.

Besides passing e-memes, you can also annotate or elaborate on them. In this case, your observation becomes linked to the e-meme it comments on, and when the e-meme is sent to another PDA, these links are preserved so that you really can read and add to threads of thought.

At this

So are lava lamps and bell-bottoms.

The theme of our conference is "retro," but the technologies are up-to-the-minute.

TECHNICAL



SAN FRANCISCO 94

That's why software designers, developers, technical coordinators, device driver developers, LAN specialists, MIS managers, consultants and training executives definitely shouldn't miss it.

Improve your productivity with sessions on OS/2, LAN systems, graphics, OOP, multimedia, pen, database and communications. Learn to write device drivers for displays, printers, storage, LANs and input devices.

bugs are



software conference,

Everyone is invited to experiment in our OS/2 and LAN lab, and see exhibits from

See the latest PSP technologies, April 25-29, 1994
Now including the Device Driver Conference

software vendors who exploit PSP products.

Hear from IBM's own Lee Reiswig, Jr., President of the Personal Software Products Division.

cool.

Also, enjoy premiums, raffles, a "special event" and more. For more information or to sign up, call 1 800 872-7109

(USA and Canada).*

It's gonna be groovy.

Operate at a higher level.™

*Outside the USA and Canada, call 1 508 443-4990. Volkswagen, the Volkswagen logo and "Beetle" are registered trademarks of Volkswagen, AG. Used by permission of Volkswagen, AG. IBM and OS/2 are registered trademarks and "Operate at a higher level" is a trademark of International Business Machines Corporation. © 1993 IBM Corp.

(continued from page 74)

Anonymity and Privacy

Have you noticed that Internet and FidoNet messages often arrive signed by everybody who handled them along the way? There are some interesting personal privacy implications if you extend this networking custom to an EchoNet. For example, if I get a message that has a path from Bob to Jim to Sally to Pete, then I could try to deduce that Jim was in the vicinity of Sally at some time. Due to name spoofing and path hacking (not to mention people borrowing each other's PDAs), this isn't ironclad evidence, but it is providing some information about the whereabouts of both your PDA and, by association, yourself.

Fortunately, EchoNet can forgo this cyberspace territorial-marking custom. There is nothing in the EchoNet relay algorithm that requires knowledge of how the message got to your PDA or the fact that it ever went through it. You can participate in EchoNets completely anonymously, or under a pen name. EchoNet pen names are like the handles of CB radio and the nicknames of Internet Relay Chat, with the advantage that you don't ever have to use an FCC-approved call sign or NIC-approved Internet address.

Privacy on an EchoNet is a little more problematic. If I can't understand what the message says, it's hard for me to figure out if it's for me or not. If it's encrypted and I don't have the key, then I'm pretty sure it isn't. Furthermore, if I can't read the message, then I can't determine if I want to pass it on or not and probably won't. Therefore, since encrypted e-memes will probably die out faster than unencrypted ones, I'd expect to see a resurgence of the clear text forms of encoding. This leads you to wonder which properties of an e-meme make it travel the farthest.

An EchoNet Application: Measurements and Surveys

Psychologists who study cliques have devised fascinating ways of measuring the who-knows-whom connectivity between people. In one experiment, you're given a booklet describing a target person. This description does not include his or her name or whereabouts. You're asked to enter your name and address in the booklet, then pass it to somebody you know who stands a chance of getting the booklet to the target. The person to whom you give the booklet repeats the process, and sooner or later the booklet ends up in the hands of the

person it describes. By counting the names in the booklet when it arrives, the psychologist obtains an upper-bound estimate of the who-knows-whom distance between you and the target. These experiments are called "studies of the small-world problem."

Suppose the people receiving the booklet had also been requested to enter some other information about themselves, rather than just entering their names and addresses. Then the booklets would accumulate a survey of all the people who handled them. Now suppose that the people are PDAs and the booklets are e-meme threads. What you have is a low-cost way of taking measurements and surveys.

For measurements, specially equipped PDAs can operate completely autonomously. At regular time intervals, the measurement's sensor is read, and a time- and location-stamped sensor value is queued as an outgoing e-meme. Over time, some of these readings find their way back to the person interested in them. While the coverage both in time and space is unpredictable, expenses are kept to a minimum and the flow is continuous.

An e-meme survey is more in the spirit of EchoNet. In this case, your PDA re-

New TestCenter™ helps UNIX® C and C++ programmers test their code more thoroughly than any other tool—resulting in substantially higher quality applications. While other testing tools keep you in the dark causing you to miss costly errors, **TestCenter's** graphical test coverage tells you which parts of your code haven't been tested—so you can see where additional testing

is needed. **TestCenter** is extremely easy to use—run-time error checking and memory leak detection are automatic. And, by testing so completely, you'll set new quality standards. Call today for your free evaluation software of the most comprehensive testing tool designed specifically for programmers.

Free Eval Software



TESTING YOUR CODE WITHOUT TESTCENTER IS LIKE SHAVING IN THE DARK.

CENTER LINE™

Programming Tools that Make a Difference™

1-800-NOW-CNTR

10 Fawcett Street, Cambridge MA 02138 (617) 498-3000 (Formerly Saber Software). email: info@centerline.com ©1994 CenterLine Software, Inc.
All company and product names are trademarks of their respective companies.

CIRCLE NO. 742 ON READER SERVICE CARD

Subscriber Service

In order for **Dr. Dobb's Journal** to provide you with the best in Subscriber Service, we have compiled the listing below to help answer any of your service related questions. Please clip and save for easy reference.

SUBSCRIPTION SERVICE

For all subscription inquiries regarding billing, renewal, or change of address: **Call Toll-Free 1-800-456-1215** for U.S. and Canadian subscribers. **Call 303-447-9330** for all other subscribers. Your mailing label will come in handy when speaking with our Customer Service Representatives.

To order new or gift subscriptions, please send your request to: Dr. Dobb's Journal, P.O. Box 56188, Boulder, CO 80322-6188.

MOVING?

Please try to give us four to six weeks notice to ensure uninterrupted service. Subscriptions are not forwarded unless requested. Be sure to include your old address, your new address, and the date you'll be at the new address. Attach your mailing label showing your old address and account number - this is always helpful. Send to the above address.

DUPLICATE COPIES?

Duplicated copies can occur when there is a slight variation in your name and address. Please send both mailing labels when notifying us of duplicates. Be sure to tell us which address you prefer.

JUST MADE A PAYMENT - BUT STILL RECEIVING BILLING AND RENEWAL NOTICES?

A notice could have been generated just prior to your payment. If you have just made a payment, please ignore recent notices. It is most likely they have crossed in the mail.

MAILING LISTS

From time to time we make our subscriber list available to carefully screened companies whose products may be of interest to you. If you would rather not receive such solicitations, simply send us your mailing label with a request to exclude your name.

So many
industry leaders
agree on
one object database,



Scott McNealy
Chairman and CEO
Sun Microsystems

Ray Noorda
President and CEO
Novell, Inc.

Steve Mills
General Manager
IBM Software Solutions

isn't it worth
a phone call?
1-800-962-9620

It sure is. Because when you call Object Design®, you'll get the inside story on the object database from a company Ray Noorda called "*the driving force behind the future of ODBMS,*" Scott McNealy described as "*the foundation for a new generation of enterprise-wide applications*" and Steve Mills declared is "*the leader in object database technology.*"

It's called ObjectStore®, and discovering what these leaders already know is easy. Just call for your free Object Database Evaluation Kit, featuring an Evaluation Guide, user stories and our video.

With the Kit, you'll learn what to look for in an object database. You'll see the importance of interactive performance, scalability and persistence independent of type. And you'll discover how our innovative virtual memory mapping architecture delivers those benefits, making ObjectStore the world's leading object database.

It's all in the Kit. Along with details on exclusive features. So if you're ready to choose an object database, call for your Evaluation Kit now. And see what everyone is talking - and agreeing - about.



Get your free Kit now!

**1-800
962-9620**
Or e-mail
info@odi.com

Object Design, Inc.
Corporate Headquarters
Burlington, MA USA

Wiesbaden, Germany: 49-611-97719-0
Swindon, UK: 44-793-486111
Sydney, Australia: 61-2-212-2766
Tokyo, Japan: 81-3-3263-9870
Boulogne, France: 33-1-46-99-13-00


OBJECT DESIGN
Leadership by Design®

©1994 Object Design, Inc. Object Design and Leadership by Design are registered trademarks and ObjectStore is a trademark of Object Design, Inc.

(continued from page 76)

ceives a questionnaire as the head of a thread of responses. The thread head asks you to append the thread with your response to the questionnaire. You're free to throw the whole thing away or look at the responses of other people before you add your own. As with PDA measurements, we don't exactly have a controlled experiment, but then, we don't have to bear the cost of conducting one, either.

A downside of e-meme surveys can be getting the raw data back to the person conducting the survey. If the PDAs are moving around, you cover a wide area but may only get back a portion of the measurements you took. On the other hand, if the PDAs are relatively immobile, you'll cover less area but stand a better chance of collecting more data. In this case, you can take advantage of the fact that the PDAs are immobile and upgrade the basic EchoNet protocol to include a notion of routing.

What if, in sending around passive text fragments, there were some way of sending executable code fragments? I think this is what people have in mind when they talk about Knowbots and General Magic Telescript agents. If there were some way of telling friendly virus-

es from evil ones, then a code fragment that hopped from PDA to PDA, gathering up data and then heading home at the end of the day, would be a terrific way to cover a lot of territory quickly. If nothing else, the quitting-time algorithm will be fun to design.

EchoNet Routing

What if our PDAs aren't roaming around but are sitting still, in a classroom, for example, or at a concert or in an office? Here, rather than trusting to random passoffs to get messages through, the PDAs can run a routing algorithm so that each PDA knows exactly which PDAs are out there and which PDAs a message has to go through to get to a particular PDA.

The simplest routing algorithm begins by each PDA figuring out who it is directly connected to. It does this using the usual message-exchange protocol, but rather than exchanging messages, it exchanges connectivity information. "Curly here. Anybody out there?" one says, and gets back a bunch of messages: "Yeah, Sleepy here. What's up?" "Yeah, Sneezzy here. What's up?" "Yeah, Doc here. What's up?" "Yeah, Bashful here. What's up?" Now Curly knows he's directly connected to Sleepy, Sneezzy,

Doc, and Bashful, and each of these knows they are directly connected to Curly.

With this information, Curly can address messages directly to particular PDAs rather than broadcasting them to everybody and having to deal with all their responses. "Curly here. Who are you connected to, Sleepy?" Sleepy responds to Curly, "Sleepy here. I'm connected to Doc and Snow White, Curly." "Ahh! A new player," thinks Curly. What Curly has discovered is that there is a Snow White out there that he can get a message to by way of Sleepy. "Curly here. Give this to Snow White, Sleepy: 'Yo, SW, what's cookin?'"

What you have here is explicit routing. Rather than just broadcasting a message into the ether, Curly sent it directly to Sleepy along with explicit instructions to pass it directly to Snow White. We've also added the peer-to-peer "Who are you connected to?" message. By coupling this message with direct addressing, any PDA can discover the entire known universe and its connectivity. Knowing this, your PDA can present you with a list of all the PDAs with which you can communicate on EchoNet and can send any message directly to the one you pick. In a sense,

GRAF/DRIVE PLUS 4.0

Turbo Graphics for Printers and Plotters

New Version!

Publication-quality hard copy graphics for Turbo C & Pascal, with minimal change to your screen graphics code. We write BGI drivers that attach to the standard C/Pascal graphics library. You'll see our drivers in commercial software for desktop publishing, business charts, investment analysis, maps, and science and engineering.

- LaserJet, Epson, DeskJet, DeskJet 550C, Canon Laser, PostScript, ProPrinter X24, PaintJet, HPGL plotters, and more.
- PCX, DXF, WPG, WMF, color dot matrix, and more (DTP Drivers).
- GUI friendly—unlike our competitors, we don't clear your screen.
- Portrait or landscape, any size from 1 inch to 5 feet.
- Not a screen dump. You get the full printer resolution.
- Not TSR. Loads on the heap at runtime. EMS optional.
- Arc, bar3d, circle, fillpoly, line, ellipse, pieslice, line/fill patterns, etc—all 100% interchangeable with Turbo's screen graphics. Also putimage, with some limitations. Plot to LPT1-4, COM1-4, or disk.
- Print PCX files, with scaling and dithering.
- Print spooler.
- Use PostScript fonts and LJ soft fonts on appropriate printer, plus Borland stroke fonts and default font on any device.
- "Highly recommended"—Jeff Duntzman, Dr. Dobbs.

For Turbo C/C++, Turbo Pascal 5-7 real mode, and Pascal 7 DPML. Personal License \$149, Developers with royalty-free distribution \$299. DTP Drivers \$99. 30-day m/b guarantee.

FLEMING SOFTWARE BOX 569 OAKTON, VA 22124

(703) 591-6451

TURN YOUR PC INTO A SOPHISTICATED, EASY TO USE, SERIAL COMMUNICATIONS PROTOCOL ANALYZER

"MAKE THE CONNECTION" ... \$395⁰⁰

- CAPTURE & TRANSMIT DATA
- EGA/VGA FONT REPLACEMENT
- USES COM1 & 2, RTX485 OR RTCARD



V1.01 \$395 V2.01 \$495

PROGRAMMABLE USING BUILT IN "C" COMPILER AND EDITOR

- DATA TO & FROM DISK
- μSECOND TIMING
- HEX/ASCII EBCDIC
- PRINTF

RTCARD™

INTELLIGENT SERIAL COMMUNICATIONS \$395⁰⁰

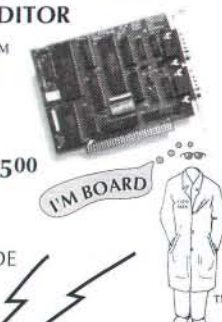
- SYNC / ASYNC • 80C31 μP
- 32K RAM • 85C30 MPCC

USE OUR DRIVER SOFTWARE ... OR DOWNLOAD YOUR OWN CODE

RTX485

RS-232 TO RS-422/485 CONVERTER

\$195⁰⁰



REALTIME CONTROL, INC.

P.O. Box 140370
Gainesville, FL 32614-0370
(904) 373-2626

(800) 232-0485

WE'VE EXAGGERATED HOW MUCH CALCULATING POWER IS IN NEW MATHCAD PLUS 5.0. BUT ONLY SLIGHTLY.

It gives you more advanced math capability than ever before. It lets you tackle harder problems and solve even tougher equations. In short, it's the most powerful, most advanced version of Mathcad® ever released.

And that's no exaggeration. More powerful than spreadsheets or calculators, easier than programming languages, new Mathcad PLUS 5.0 gives engineers, scientists and educators more tools to do calculations with greater speed and ease.

You get more functionality for computing derivatives and integrals, differential equations, advanced vector and matrix operations, statistical functions, curve fitting, and fast Fourier and wavelet transforms. You can choose from a wider range of symbolic capabilities, and

graph in 2-D and 3-D polar, contour and parametric plots. MathSoft's Electronic Books, based on the most popular reference books, let you instantly cut and paste

Mathcad PLUS 5.0's on-screen worksheet lets you easily combine equations, text and graphics.

hundreds of formulas into your work. And with Mathcad PLUS Function Packs you can add even more remarkable

calculating power in specific disciplines like signal processing, data analysis, statistics and graphics.

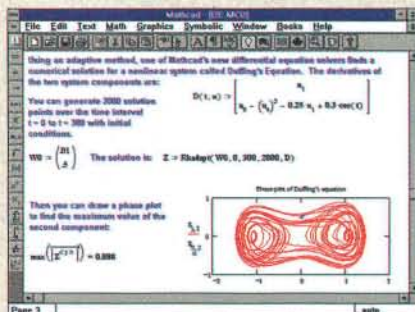
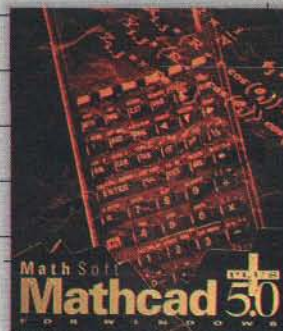
Plus, like its Mathcad predecessors, it's as easy and intuitive as using a scratchpad. Simply enter equations in real math notation anywhere on Mathcad PLUS 5.0's on-screen worksheet. Add text and graphics. Change variables and instantly update your work. Mathcad PLUS 5.0 calculates answers quickly and accurately, then prints your results in impressive, presentation-quality documents.

So try Mathcad PLUS 5.0 today, and tomorrow you'll be ten times more productive. Well, maybe we're exaggerating again. But only slightly. Mathcad PLUS 5.0 is priced at just \$299⁹⁵. To order, or to receive even more information, call 1-800-967-5075. Or mail or fax the coupon below (Fax: 716-873-0906).

FREE MATHCAD PLUS 5.0 INFORMATION KIT

For more information on Mathcad PLUS 5.0, mail or fax this coupon.

Name	Title	a22a2
Company		
Address		
City		
State	Zip	
Country	Phone	



(continued from page 78)

your PDA is functioning as a router or a switch as well as a source, sink, and passive relay point. You're realizing some of the advantages of a switched network without building a central switch through which all traffic must flow.

There are hundreds of network-discovery and network-routing algorithms and protocols. Many can be used in EchoNets and switched nets. In fact, an EchoNet can be thought of as just a network, in which every node is also a router. Recently, the Internet technical community has become interested in supporting Internet connectivity to mobile hosts (see the accompanying text box entitled "Mobile Internetworking") and has published the "Internet Packet Transmission Protocol," which discusses a possible routing algorithm for this situation.

EchoNets as Distributed Systems

This primitive network-discovery and routing algorithm is an example of a large class of distributed-system algorithms that has received attention over the last 15 years. Dijkstra's classic paper [Dijkstra 80] and Chang's independent discovery [Chang 82] have set the tone and direction for much of this work.

Chang's paper is a more readable introduction to distributed algorithms even though Dijkstra's paper has priority. Yang and Marsland's recent note [Yang 93] is an excellent annotated bibliography on two important problems in this field.

Dijkstra and Chang showed that it is possible to design practical algorithms for EchoNets which enable any node in the network to discover information about the whole network or about any particular node in the network. In fact, Chang called these algorithms "echo algorithms" because a broadcast question produces an echoed response. "Practical" here means that the answer is obtained in a deterministic and computable amount of time and that the EchoNet message traffic generated by the broadcast request eventually dies out.

The Dijkstra/Chang echo algorithm proceeds as follows: The initial, inquisitive node sends its question to each of the nodes to which it is connected. Upon initially receiving the question, each node relays the question to all the nodes to which it is connected except for the initial node. If the receiving node has no other nodes to which it can send the question, then it sends the accumulated answer back to the node that sent

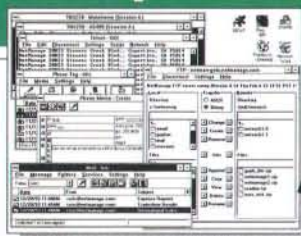
it the question. Finally, when a node receives answers from all the nodes to which it sent the question, it in turn sends the accumulated answers to the node that first sent it the question.

In his paper, Chang gives a number of applications of this basic echo algorithm along with some performance calculations and special-case improvements. One of the applications, the Single-Source Sort, is particularly applicable to our PDA-to-PDA communication situation. The idea is that a new node is joining an existing EchoNet and wants to pick a unique identity. The new node sends out the question, "What is your name?" Each node's echo is its own name, appended to the list of names it has received from the nodes it has contacted. What arrives back at the new node is a list of the names of all the nodes in the EchoNet. All the new kid on the block has to do now is pick a name that isn't on the list.

Global State and Cooperative Behavior

Dijkstra/Chang-style echo algorithms are fine for determining static properties of an EchoNet (such as the list of the names of all the nodes in the network); but what about dynamic properties? Suppose all the nodes in an EchoNet

TCP/IP for Windows



New Version!
CHAMELEON 4.0
AUTOMATIC
INSTALL OVER
ODI OR NDIS

More Windows applications than any other TCP/IP package
Implemented as 100% Windows DLL (not a TSR)
Requires only 6KB of base memory
Installs in 5 minutes

- Works concurrently with NetWare, LAN Manager, Vines, Windows for Workgroups, etc.
- Up to 128 simultaneous sessions

Developer Tools:

Windows Socket API
Berkeley 4.3 Socket API
ONC RPC/XDR
WinSNMP API

Applications:

TELNET (VT100, VT220, VT),
TN3270, TN5250, FTP, TFTP, SMTP
Mail with MIME, News Reader, PROFS
Mail, LPR.LPD, Ping, Bind, Finger,
Whois, Gopher, Phonetag, Scripting,
Statistics, Custom, SNMP Agent

NEW!

Gopher Client, TN5250, LPR/LPD
MIME Support in Mail, Scripting



For overnight delivery call:

NETMANAGE™
(408) 973-7171

20823 Stevens Creek Blvd.
Cupertino, CA 95014 USA
Fax (408) 257-6405

CIRCLE NO. 805 ON READER SERVICE CARD

PUZZLED BY WINDOWS™?

ARE YOU TIRED OF
LOSING PRECIOUS TIME
AND MONEY? LEARN THE SECRETS
OF MICROSOFT WINDOWS™ FROM THE
INDUSTRY'S MOST EXPERIENCED WINDOWS™
EXPERT TRAINERS. ON SITE COURSES AND PUBLIC
TRAINING OFFERED WITH INVALUABLE HANDS ON
LAB TIME. ISD — THE PAUL YAO COMPANY

800-942-3535

International
Systems
Design

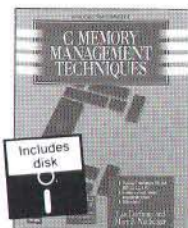
1075 BELLEVUE WAY NE, SUITE 300 • BELLEVUE, WA 98004-4276 • 206-828-6402

CIRCLE NO. 895 ON READER SERVICE CARD

4 books for only \$4⁹⁵

Values to
\$183.00

when you join the *Computer Professionals' Book Society*



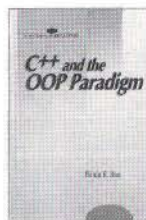
4191P \$32.95



9309P \$29.95



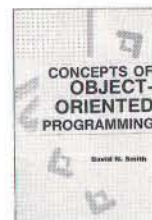
881933P-XX \$39.95
Counts as 2



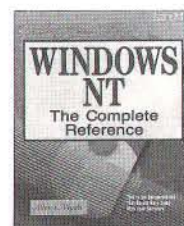
051140H \$40.00
Hardcover



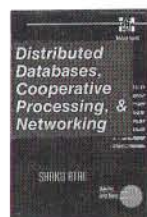
016732H \$40.00
Hardcover



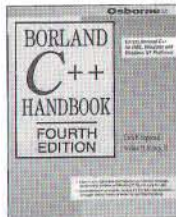
10039P \$24.95



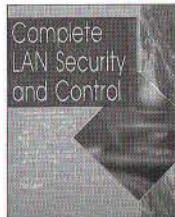
881832XP-XX \$29.95
Counts as 2



157673H \$50.00
Hardcover



8819601P-XX \$34.95
Counts as 2



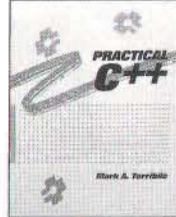
4490H \$39.95
Hardcover



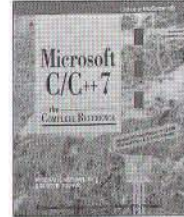
0566933H-XX \$70.50
Counts as 2/Hardcover



3484H \$34.95
Hardcover



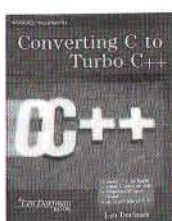
0637385H-XX \$39.95
Counts as 2/Hardcover



881664P-XX \$29.95
Counts as 2



8819520P \$39.95



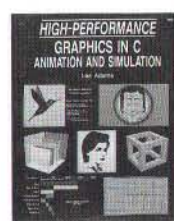
4084H \$39.95
Hardcover



4114P-XX \$39.95
Counts as 2



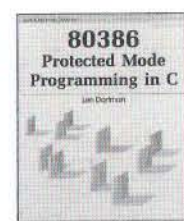
4340P \$34.95



3049P-XX \$29.95
Counts as 2



881796P \$27.95



3736P \$24.95



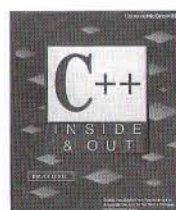
006215H \$40.00
Hardcover



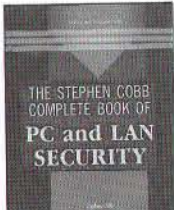
881653P-XX \$29.95
Counts as 2



4103P \$24.95



881809P-XX \$29.95
Counts as 2



3280H \$36.95
Hardcover



005076H-XX \$50.00
Counts as 2/Hardcover



881691P-XX \$39.95
Counts as 2



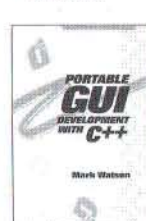
4341P \$34.95



881654P \$29.95



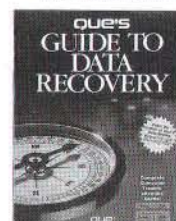
586118H-XX \$39.95
Counts as 2/Hardcover



0684898H \$45.00
Hardcover



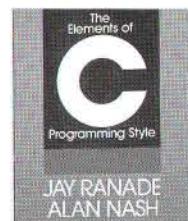
006551H \$49.00
Hardcover



586104P \$29.95



051216H-XX \$39.95
Counts as 2/Hardcover



051278P \$29.95



4196H \$32.95
Hardcover



3489P \$24.95

As a member of the Computer Professionals' Book Society...

you'll enjoy receiving Society bulletins every 3-4 weeks containing exciting offers on the latest books in the field at savings of up to 50% off of regular publishers' prices. If you want the Main Selection do nothing and it will be shipped automatically. If you want another book, or no book at all, simply return the reply form to us by the date specified. You'll have at least 10 days to decide and if you ever receive a book you don't want, due to late mail delivery of the News, you can return it at our expense. And you'll be eligible for **FREE BOOKS** through the Bonus Book Program. Your only obligation is to purchase 3 more books during the next 2 years, after which you may cancel your membership at any time.

DDJ494

All books are softcover unless otherwise noted. Publishers' prices shown. If you select a book that counts as 2 choices, write the book number in one box and XX in the next. A shipping/handling charge & sales tax will be added to all orders. ©1994 CPBS

If card is missing, write to:

Computer Professionals' Book Society, Blue Ridge Summit, PA 17294-0870

(continued from page 80)

wanted to cooperate in accomplishing a task of some sort. How would they keep track of the current state and progress of their combined effort, or stay coordinated?

One way would be to have everybody synchronize their actions to a global clock, then treat the dynamic state as simply a series of static states separated by network-wide time synchronization points. From an individual node's point of view, the drill might look something like this (see, for example, [Flamner 92]):

1. Do something useful.
2. Wait until the global-synchronization point.
3. Exchange what you've done with everybody else and find out what everybody else has done using an echo algorithm.
4. Figure out what to do next.
5. Go to step #1.

While there are a number of global-clock and virtual-time algorithms which can be used for the global-synchronization point [Yang 93], you get the feeling there's an excessive amount of overhead in this approach—there must be a less West Point and more Mill Valley way of achieving cooperation.

Chandy and Lamport [Chandy 85] describe an algorithm whereby nodes in an EchoNet can determine the global state of a cooperative effort without a global clock. They called their algorithm a "distributed snapshot" by analogy to a group of photographers (the nodes) who take several pictures (the local states) and piece together the results (by exchanging messages) to form a "meaningful" panoramic picture (the global state) that is larger than a picture that any one photographer's camera could handle. In this context, "meaningful" means that the composite picture is sufficient for coordinating the nodes and getting the cooperative effort accomplished.

The Chandy/Lamport algorithm provides a method for "strobing" the recording of a node's local state by a mechanism other than a global alarm clock. The method is based on the sending of a special "Record your state!" message around the network. After the message has been received and obeyed by all nodes, the recorded local states are collected to form a description of the global state; this global state description is distributed to all nodes using an echo algorithm.

Since the "Record your state!" message reaches nodes at different times,

you have to record not only the state of each of the nodes but the state of what causes nodes to change state; videlicet, the in-transit message traffic between nodes. The resulting global state is rather like a little film clip that we can run forward and backward to see what the network was up to during a tiny interval of time. The Chandy/Lamport algorithm is a careful specification of how the local states of the nodes and the communication channels are to be recorded so that this movie is a useful representation of the net's state.

The Chandy/Lamport distributed snapshot algorithm works like this:

Initiation Rule: Send the "Record your state!" message to each node to which you're connected and then record your state before you send any further messages.

Unrecorded State Rule: If you receive a "Record your state!" message and you have not already recorded your state, then: 1. Record your state; 2. record the fact that the state of the channel between you and the node that sent you the message is "Empty"; 3. start recording all subsequent messages you receive from other nodes; and 4. send the "Record your state!" to each node to which you are connected.

Recorded State Rule: If you receive a "Record your state!" message and you already have recorded your state, then record the fact that the state of the channel between you and the node that sent you the message is the sequence of the messages you got from this node between the time you recorded your state and the current "Record your state!" message.

The algorithm gives receivers the obligation of recording the in-flight messages. The recording starts at a node

Mobile Internetworking

A number of similar schemes (Ioannidis, Uehara, and Wada, for example) have been proposed for extending TCP/IP, and hence, the Internet, to mobile computers. The situation is a little more complicated because, as originally conceived, TCP/IP addresses combine two distinctly different pieces of information: unique name and current location (kind of like "Minnesota Fats" or "Boston Blackie"). When host computers were immobile, this didn't matter; if they did move, we changed their names. Clearly this solution won't work for a computer tooling down Route 66.

A design criterion for all of the mobile-internetworking proposals is to minimize the impact of supporting mobile hosts on the existing network as much as possible. Thus, not only should everything that works today continue working, but a stationary host should be able to communicate with a mobile host just as if it were another stationary host. Basically, this means that the mobile host's address doesn't change, at least from the point of view of other hosts communicating with it.

The Internet Packet Transmission Protocol (IPTP) proposed in a July 1993 Internet Draft (Wada) endows a mobile host with two addresses: a home address (the unique name, "Blackie") and an away address (the current location, "Boston"). The home address doesn't change and is the permanent name of the host known to the world. The away address does change and is the address at which the mobile host

can currently be reached. The only change to the network is the addition of a piece of software called a Packet Forwarding Server to the mobile host's home network, which keeps track of where the mobile host is and forwards messages to it.

As the mobile host moves around, it acquires an away address from each local network whose territory it enters, and it sends this away address back to its home network's Packet Forwarding Server. In this way the home network always knows where the mobile host is and how to reach it. Messages to a mobile host are always sent to its home address; when they're received, the Packet Forwarding Server readdresses them to the mobile host's away address. Messages from a mobile host go directly to whom they are addressed and need not detour through the Packet Forwarding Server on the mobile host's home network. The return address on these messages is the mobile host's home address, not its temporary away address.

As hosts and routers on the Internet are willing to become more mobile-host-aware, there are a number of efficiencies that can be introduced into this minimum-impact protocol, and many of these are described in the referenced papers. For example, a sender might indicate that it is willing to track the mobile host as well so the mobile host could set its return address to its away address rather than its home address.

—S.B.G.

Gates Motel

NO VACANCY

No one has yet been able to match our performance. No wonder QEMM outsells all the others *put together!*

QEMM™ puts the maximum memory right where your favorite programs need it so you

***"I needed another 32K for my favorite TSR.
I added 2 megabytes. I still need 32K! What gives?"***

can run them and TSRs without 'out of memory' problems or conflicts. That keeps your PC running smoothly and performance at its best.

Whether you're running MS-DOS, IBM[®] PC-DOS, DR DOS, Novell DOS or MS Windows; one megabyte or eight, don't sacrifice; don't compromise; don't risk losing work.

Managing your memory well is the best way to assure your work won't go to waste.

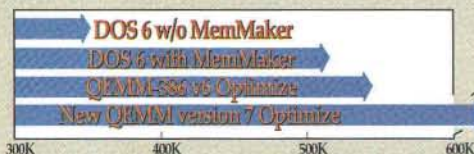
QEMM version 7 is the most powerful, flexible memory manager you can buy.

QEMM comes with the new version of Manifest, the award-winning memory analyzer that helps you see how your PC works.

It's the utility that finds memory
when nothing else can.

VACANCY

The more memory you have, the more flexibility and reliability you can enjoy. Thanks to



We tested DOS with and without MemMaker and with QEMM 6 and our new QEMM 7 runs away from all of them. See details of test conditions below.

**Protect Your Productivity;
Keep Your Work Safe.**

Any task, from programming to writing the company business plan to composing a personal letter, takes time and thought. Your PC is supposed to make that process easier; your output better. When you can't run your favorite grammar-checking TSR or have to get by without a vital network utility, you're sacrificing productivity.



Quarterdeck

Quarterdeck Office Systems, 150 Pico Boulevard, Santa Monica, CA 90405 (310) 392-9851 Fax (310) 314-4219

How we got the client system: CPU = 486/33; All R/Power Business VESA monitor supported with 16 steps of RAM and running MS-DOS 6.0 Comparisons were done using the following memory managers: UMM7, CEMM32, MZ-CXK & MemMaker. In addition to the driver(s) required by each memory manager, the following drivers, DOS resources and programs were loaded for all comparisons: INTCARD, SERIAL-IDE, IDE-HIGH, IDE-LOW, EISA-BUS, LAN-SUPERMAN, SUPERMAN-SERVERS, TFS-FILE, JEDNET, PCL-ALPHATEC, BETA-VGA, MATHY, INDEXER, S-LINK, MEDICAL, POWER-CAM, NETS-INNOVIX, MYSOFT, CAM-SMARTVIEW, COM-INTERACOM, COMI customers Office Systems. The authors are grateful to their associates.

CIRCLE NO. 919 ON READER SERVICE CARD

(continued from page 82)

when the first "Record your state!" message appears on any channel and stops channel-by-channel when "Record your state!" appears on each channel.

Extended Realities

What the Dijkstra/Chang and Chandy/Lamport algorithms give us is a way for many mobile computers to act cooperatively. While each individual PDA is keenly aware of its own surroundings, it can also count on the "eyes and ears" of the other PDAs in its EchoNet to act as lookouts in regions beyond its own ken. I think of PDAs knitted together by these algorithms as being similar to the compound eye of an insect or a very large array radio antenna. In a sense, the reality of each PDA has been extended to the area covered by the entire EchoNet of which it is a member.

It's interesting that this relatively complex form of network behavior has been achieved without a central switch. Switchless networks like EchoNets certainly have their drawbacks, such as indeterminate message delivery. The advantages, however, include robustness due to absence of a single point of failure and the ease with which nodes can enter and leave the communication

mesh. In the era of mobile wireless computing, we may find situations where it just doesn't make sense to send the message downtown and back if it only has to get to someone standing next to me. We may also find useful forms of network communication that don't send us a bill at the end of the month.

Bibliography

Acharya, Arup, and B.R. Badrinath. "Delivering Multicast Messages in Networks with Mobile Hosts." *Proceedings of the 13th International Conference on Distributed Computing Systems*. May 25-28, 1993, Pittsburgh, PA.

Bush, Randy. "FidoNet: Technology, Tools, and History." *Communications of the ACM* (August 1993).

Chandy, K. Mani, and Leslie Lamport. "Distributed Snapshots: Determining Global States of Distributed Systems." *ACM Transactions on Computer Systems*, (February 1985).

Chang, Ernest J.H. "Echo Algorithms: Depth Parallel Operations on General Graphs." *IEEE Transactions on Software Engineering* (July 1982).

Chow, Ching-Hua. "On Multicast Path Finding Algorithms." *Proceedings of IEEE Infocom '91*.

Dijkstra, E.W., and C.S. Scholten. "Ter-

mination detection for diffusing computations." *Inf. Proc. Lett.* (August 1980).

Ioannidis, John, and Gerald Q. Maguire, Jr. "The Design and Implementation of a Mobile Internetworking Architecture." *Proceedings of the 1993 Winter USENIX Meeting*, January 25-29, 1993, San Diego, CA.

Flammer, George H. *Method for Synchronizing a Wide Area Network without Global Synchronization*. U.S. Patent #05130987.

Ioannidis, John. *Protocols for Mobile Internetworking*. Ph.D. Thesis, Columbia University, 1993.

Trask, Jeremy R., and Anthony Wiener. "Data Communication System." U.S. Patent 4,937,569, June 26, 1990.

Uehara, Keisuke, et al. "Enhancement of VIP and Its Evaluation." *Proceedings of INET '93*, August 17-20, San Francisco, CA.

Wada, Hiromi, Tatsuya Ohnishi, and Brian Marsh. "Packet Forwarding for Mobile Hosts." Internet Draft, July 1993.

Yang, Zhonghua, and T. Anthony Marsland. "Annotated Bibliography on Global States and Times in Distributed Systems." *ACM Operating Systems Review* (July 1993).

DDJ

To vote for your favorite article, circle inquiry no. 8.

ADD THE POWER OF GRAPHICS TO YOUR PROGRAMS WITH GENUS

GENUS
MICROPROGRAMMING

1155 DAIRY ASHFORD • SUITE 200 • HOUSTON, TX 77079 • (713) 870-0737
FAX (713) 870-0288 • BBS (713) 870-0601

TO PLACE YOUR ORDER OR FOR A FREE GENUS CATALOG AND DEMO DISK, CALL TOLL-FREE:

1-800-227-0918

Supporting the myriad of graphics adapters and printer devices available can be a real nightmare, and getting different toolkits from different companies to work together in your program is a big hassle. But not anymore!

Genus offers a one-stop solution to your programming needs with its complete line of programming toolkits. Designed to work separately or work together, the GX Development Series allows you to incorporate graphics into your programs quickly and easily, saving you time and money. Not only are the toolkits easy to use, they are written in 100% assembly language for the smallest size and fastest code possible.

Find out what thousands of professional programmers already know...

For comprehensive graphics support, you need the GX Development Series!

GX GRAPHICS is a complete graphics library supporting all graphics primitives from Pixels to Ellipses. **ONLY \$249**

PCX TOOLKIT allows you to create, display, and manipulate PCX images from almost any program. Available for DOS or Windows. **ONLY \$249**

GIF TOOLKIT allows you to create, display, and manipulate GIF images from almost any program. **ONLY \$249**

GX EFFECTS lets you add special effects, animation, and sound to your programs. Available for DOS or Windows. **ONLY \$199**

GX TEXT lets you display bitmapped text and stroked fonts in any graphics mode as simply as text mode and includes 600+ fonts. **ONLY \$149**

GX PRINTER provides comprehensive graphics printer support for your programs (over 300 printers supported). **ONLY \$349**

GX IMAGES allows you to import and export bitmapped graphics in a wide variety of file formats. Available for DOS or Windows. **ONLY \$495**

PROTEUS allows you to create demos, tutorials, and prototypes quickly and easily. **ONLY \$349**

DEVELOPER'S PAK allows you to save 20% when you order 3 or more Genus toolkits. **CALL FOR DETAILS**

- No Royalties!
- 30-Day Money-Back Guarantee
- Assembly Language Speed and Size
- Comprehensive Manuals and Examples
- Hercules/CGA/EGA/VGA/SVGA Support Through 1280x1024x16.7M Colors
- Supports 8 Languages and Over 50 Resolutions, Plus Mode X
- Free GX Kernel library provides scaling, color conversion, and EMS/XMS/Disk memory
- Real Mode, Protected Mode, and Source Code Versions Available
- Also Available From Genus... **GX GAMES \$59**

©1993 GENUS MICROPROGRAMMING, INC. 5069

DESQview/X

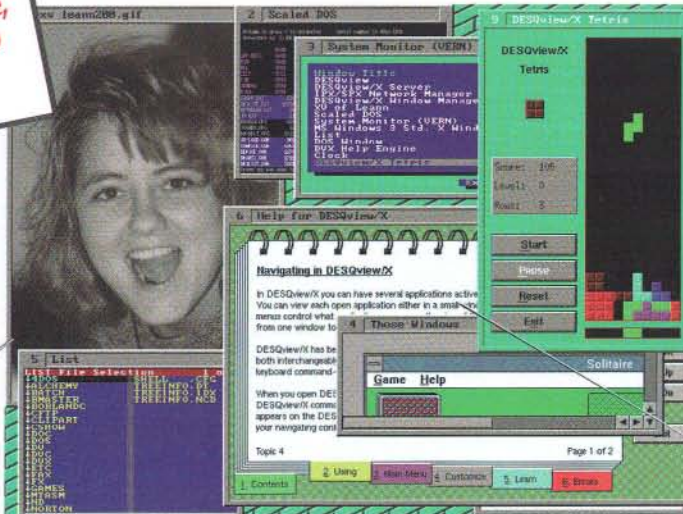
The simplest way to integrate DOS, MS Windows and X Window software on your PC—or any other PC on a network.

New Version 1.2 includes enhanced network support: PC-NFS, Hewlett-Packard LAN Manager, Microsoft LAN Manager, Beame & Whiteside, Wollongong—in addition to PC/TCP and Novell LAN Workplace for DOS

DOS and Windows applications compatibility.

Proven memory management and diagnostics with both Manifest and QEMM.

DESQview/X has EGA, VGA, 8514A, Super VGA and DGIS high resolution graphics support.



Comfortable 'look and feel' for both keyboard and mouse users.

With scalable window capability provided by Adobe Type Manager, you can get more DOS windows on the screen than ever before.

Any DESQview/X window can be made into an icon at any time.

Online help is two keystrokes away.

Multitasking Power

DESQview is the recognized pioneer in DOS multitasking and is at the very heart of DESQview/X—giving you proven high performance multitasking of DOS and Microsoft Windows programs side-by-side.

X Window System Graphics

The X Window System gives DESQview/X its graphical interface. And its ability to access DOS text and MS Windows programs running remotely on your network. Add the optional TCP/IP Network Manager and it gives you the ability to access remote X Window programs on workstations such as an IBM RS/6000 or a SUN SPARCstation. Or vice versa.

Adobe Type Manager

The accepted standard in scalable font technology is built in so that X Window System programs can use scalable fonts. And it also gives you the ability to view DOS text programs in scaled windows.

Program Memory Management

DESQview/X has the memory management services of advanced operating systems—thanks to the built-in Rational Systems DOS4GX DOS extender (16 and 32 bit), shared libraries, dynamic link libraries (DLLs) and virtual memory.

Superior DOS Memory Management

DESQview/X comes with our award-winning, best-selling expanded memory manager, QEMM, and the highly acclaimed Quarterdeck Manifest to ensure that your PC's memory is always at its optimum and that your DOS programs have the absolute maximum memory available.

Customization

With DESQview/X's customizable menus, graphical desktop, keystroke macros, mark and transfer and online help, it is very easy to tailor the your own PC to your exact working needs. In addition, you can choose from two optional 'look and feel' packages: OSF/Motif and OPEN LOOK.

Includes Valuable Companion Programs

DESQview/X comes with three companion programs. Application Manager for launching and organizing programs. File Manager for managing local and remote files. And Icon Editor for creating icons.

Quarterdeck makes your PC a better place to work, whatever software you prefer to use.

Qty	Product	5-1/4	3-1/2	Each	Totals
	DESQview/X			\$275.00	
	DESQview/X Network Mgr. (TCP/IP)			\$200.00	
	DESQview (for 286 PCs)			\$99.95	
	DESQview 386 (for 386 and 486 PCs)			\$149.95	
Shipping & Handling \$10 (Canada & USA only)					
California Residents add 8.25%					
Grand Total					

Yes! Please Send: ☐ Visa ☐ MasterCard Expires / / Card #
 Name
 Address
 City State Zip
 Payment method
 Call (800) 354-3222 ext. 1D1 for fastest service Please allow 3 weeks for delivery

Quarterdeck

Quarterdeck Office Systems, 150 Pico Boulevard, Santa Monica, CA 90405 (310) 392-9851 Fax (310) 314-4219
 Quarterdeck International Ltd., B.I.M. House, Crofton Terrace, Dun Laoghaire Co. Dublin, Ireland Tel.(353) (1) 284-1444 Fax: (353) (1) 284-4380



Help for Windows Help Authors

Windows help authoring tools provide quick relief

Al Stevens

To be taken seriously, a Windows application must provide online help. Users have come to expect it, and developers have little choice but to provide it. However, like staff meetings, program documentation, and user's guides, it's a task that programmers approach as willingly as they would a root canal. But like it or not, most Windows developers must eventually build a help database. Fortunately, Windows includes WINHELP.EXE, an application that displays online, context-sensitive help in a standard format. You design a help database and build the hooks into the application. WINHELP does the rest.

Building a help database, however, is no easy task. If you're lucky, your boss hires a professional tech writer to do most of it. There is more to the job than writing the words and composing the pictures. You use a number of unrelated tools to convert the help words and graphics into a database format that WINHELP recognizes. You can work with these tools in their native autonomous environments or use a Help authoring tool to integrate them into a project-oriented toolset. This article describes the components of a Windows help database, addresses the manual procedures for building one, and discusses three Help authoring tools that ease the process. One tool, the Windows Help Author, comes as an unsupported application on the Microsoft Developer Network (MSDN) CD. The other two, Windows Help Magician from Software Interphase and RoboHelp from Blue Sky Software, are commercial tools from third-party vendors.

About WINHELP

WINHELP is an independent Windows application that comes with Windows.

Al is a DDJ contributing editor. He can be reached through the DDJ offices or on CompuServe at 71101,1262.

Developers use it to provide online help similar in look and feel to that of other Windows applications. WINHELP displays help text and graphics from databases that conform to a prescribed format. The format supports hypertext links, keyword searches, graphical displays and controls, and navigational controls. Applications programs associate their run-time contexts with specific topics in the Help database. The developer composes the help database, assigns run-time context identifiers to the help topics, and puts the associated context-sensitive hooks in the applications code.

A help database can have text, graphics, motion video, and sound. The text can include highlighted phrases that you click on to pop up informational windows or jump to other topics in the text. Graphical elements such as tool-bar buttons, icons, and screen shots can be displayed and clicked on. The help document can talk, display pictures, play music, and show movie clips. There is an automatic table of contents and a keyword search feature. There are navigation functions that jump forward and backward through the topics. The user can place and retrieve bookmarks in the text. All of these features are implemented by WINHELP based on a database that the developer builds.

Although WINHELP typically provides online help to applications, its hypertext, multimedia, and navigational features make it useful for presenting other kinds of information. You can run WINHELP either from within an application, or as a stand-alone program and command it to display text and graphics from any conforming database. WINHELP is commonly used for online reference and users' documentation for compilers and other applications. When you see a Program Manager group with one or more prominent yellow question-mark icons, there is a good chance that

they each run WINHELP to display a different documentation database. There are even "readme" files implemented as help databases.

Help-Project Tools and Components

Building a help database involves several tools. You need a word processor that reads and writes the so-called "rich-text format" (RTF), which is an abominable concoction of embedded ASCII tokens that define how a document should appear. The RTF editor is used to compose the help database. Theoretically, you could use an ASCII text editor to build an RTF document, but it's not advisable. You're better off using a word processor that works with the format and displays comprehensible text on the screen.

A second text editor, such as Notepad, that works with ASCII text maintains the help project file, which describes various components in the database. If your database includes graphics, you need a program to produce bitmap files. If the user is to make selections by clicking on parts of the graphics, you need the Hot-Shot Editor. The Help Compiler reads the project, text, and bitmap files and compiles the help database into the format that WINHELP expects. (The Help Compiler and Hot-Shot Editor are bundled with most Windows software-development environments and are included in the SDK.) Finally, you need your own software-development environment to put help context identifiers into the application.

Among the project components you'll use when building a help database are:

Help Project File. A help database uses an ASCII text project file with the .HPJ filename extension. The project file contains options for the Help Compiler, including the name of the .RTF text files, and the title and size of the help window. It also associates string topic identifiers

tifiers that you place in the help text with integer context identifiers in the programs.

Help Text. The help text contains the narrative text for the help database, tokens that specify the filenames and position for graphics, topic names and identifiers, and the linkages for hypertext references and keyword associations.

The RTF word processor must be able to encode double-underlined and hidden text and insert footnotes into the document, all by using RTF protocols. Footnotes, which are tagged with \$, #, and K characters and appear just ahead of the title for each topic, provide mnemonic identifiers for the topic, titles for the table of contents, and keywords for the search. Underlined text identifies hypertext phrases. Hidden text immediately follows the hypertext phrases and specifies the link's mnemonic identifier. Word for Windows 2.0 and Ami Professional both have these capabilities.

Multimedia. A help database can include graphics, sound bites, and movie clips. You include these elements by putting tokens in the text that identify what they are and their filenames. For example, to insert a bitmap you put the following token into the text at the character position where you want the upper-left corner of the bitmap to show:

{bmc dolly.bmp}

The *bmc* keyword specifies a bitmap. *dolly.bmp* is the name of the file in this example that contains the bitmap. The Help Compiler uses the token to build the graphic rendering into the database.

If the user clicks on a graphic to jump to another topic, you underline the token and add a hidden topic identifier to link to the topic. If parts of the same graphic point to different jump links, you use the Hot-Shot Editor to identify the coordinates of each jump area. For example, I put a picture of the application's tool bar in the help database and used each tool button to jump to the topic that describes its function. A weakness of the help development system is that graphics in the database slow the Help Compiler down to a crawl when you use the compression options to build the database. A big help file with many large pictures can take hours to compile.

Context-Sensitive Help

Most help databases support context-sensitive help. If they do not, the user must start at a table of contents or do a keyword search to find a particular help topic. By associating help topics in the

database with menu selections, dialog boxes, controls, and other application-specific contexts, you give the user the ability to go directly to the help topic that discusses the currently selected application context. For example, suppose that you assign context identifier 3 to the EDIT_MENU identifier. These are arbitrary values that you decide to use. In your program, you associate the integer 3 with the Edit menu label on the menu bar. In the help text, you associate the EDIT_MENU identifier with the topic in the text that explains the menu. When the user selects that menu and asks for help, winhelp.exe displays the associated topic.

To integrate a help database with an application, you modify the source code to specify the database name and to associate context identifiers with different parts of the application. How you do this depends on the programming language. Visual Basic's Options/Project menu opens the Project Options dialog box where you add the name of the help database. The Menu Design Window includes a *HelpContextID* field for the numeric value associated with the associated help topic. The Properties windows for the application's controls include similar *HelpContextID* fields.

Preventative maintenance from product inception. Quality assurance starts when the first line of code is written.

PCYACC® 5.0 LANGUAGE TOOLKIT

Includes "Drop In" Language Engines for dBASE, POSTSCRIPT, HYPERTALK, SMALLTALK-80, C++, C, PASCAL, VHDL, PROLOG, FORTRAN, FORTRAN-90, COBOL, COBOL-85, VISUAL BASIC, LEX, YACC, GW BASIC, RTF, SNA/LU, SGML, ASN, RPG, REXX, PL1, ADA, SQL, SQL2 and DB2.

PCYACC Version 5.0 is a complete Language Development Environment that generates ANSI C source code from input Language Description Grammars for building Assemblers, Compilers, Interpreters, Browsers, Page Description Languages, Language Translators, Syntax Directed Editors, Language Validators, Natural Language Processors, Expert System Shells, and Query languages.

- NEW! Generates C++ and YACC classes for Windows 3.0 and OS/2 2.0 Presentation Manager
- NEW! 32 BIT Extended Memory Support for DOS 386/486, and OS/2 2.0 will compile any size grammar
- NEW! Build professional systems with full ERROR HANDLING, RECOVERY, AND REPORTING.
- Quick Syntax Analysis and support for multiple parsers
- Lexical Analyzer generator ABRAXAS PCLEX is included

- Our grammars are 100% compatible with UNIX YACC
- Debugging tools include runtime Parser Tracking and Abstract Syntax Tree generation.

CodeCheck® 5.0 SOURCE CODE MANAGER

Includes "Drop-In" Rules for Compliance analysis, Adherence to specifications, Measures of complexity, Silent error detection, Code maintainability, and Portability.

CodeCheck Version 5.0 is a programmable tool for managing all C and C++ source code on a file or project basis. CodeCheck is input compatible with all variants of K&R, ANSI C and C++. CodeCheck is designed to solve all of your Portability, Maintainability, Complexity, Reusability, Quality Assurance, Style Analysis, Library/Class Management, Code Review, Software Metric, Standards Adherence, and Corporate Compliance Problems.

- Maintainability—CODECHECK identifies and measures complex, sloppy, and hard to maintain code.
- Portability—CODECHECK identifies code that will not port between DOS, OS/2, Unix, VMS, and the Macintosh, and check for compatibility with ANSI, POSIX, and other standards.

- Complexity—Measures program size (Halstead) and Program Complexity (McCabe).
- Compliance—CODECHECK allows your corporate coding and project specification standards to be completely automated for compliance validation.

OS	CodeCheck	PCYACC
DOS	\$495	\$495
MAC	\$495	\$495
OS/2	\$995	\$995
NT	\$995	\$995
UNIX	\$1995	\$1995
VMS	\$1995	\$1995

Educational discounts are available.

30 day Money back guarantee!

Free AIR Shipping anywhere in the world!

To Order Call **1-800-347-5214**



ABRAXAS™
Software, Inc.

5530 SW Kelly Ave., Portland, OR 97201 USA
TEL (503) 244-5253 • FAX (503) 244-8375 • Apple D2205

FAX FOR FREE DEMOS!

A C program that uses the Windows SDK API intercepts the WM_KEYDOWN message and watches for the F1 key, either from the application window's processing module or by using the *SetWindowsHookEx* function to install a filter function that intercepts messages to dialog boxes. Once the key is pressed, the program calls the SDK's WinHelp function passing the name of the help database and the context identifier number.

Microsoft Foundation Classes programmers associate help contexts with controls using the MAKEHM tool, which constructs topic mnemonics from the source-code control identifiers and assigns context identifiers to them. The mes-

sage map associates the Windows ID_HELP message to *CWinApp::OnHelp*. AppWizard builds this framework automatically when you elect to include context-sensitive help in your application.

Building a Database: The Manual Approach

The Windows Program Manager has features that help organize the tools into what approaches an integrated fashion. Recently I worked on a help database for a Visual Basic application. I wanted VB and the help-authoring tools available at the same time so that I could put context identifiers in the program while I wrote topics in the help database.

I set up a Program Manager group for the project. An icon runs Visual Basic with the application's makefile on the command line. That takes care of the software-development side of the project. Another icon runs Notepad to edit the help project file. A Word for Windows icon starts Word to edit the RTF text file. An MS-DOS prompt icon starts a DOS batch file that runs the Help Compiler. I used Paintbrush to build and change bitmaps; it has an icon in the Program Manager group. Finally, an icon runs WINHELP itself to view the help document during each stage of its development. These tools sit together as Program Items in a Program Group with the startup subdirectories and command-line document files built into their properties. Thus, not only do I avoid rummaging through all of the groups to find and run them, but they start up with the help files loaded and ready to modify.

The manual approach works, but it is not perfect. Getting into and out of Word involves telling Word each time to convert the RTF format. The help document in Word does not resemble the display that WINHELP uses. You get neither a visual tool nor WYSIWYG. To see the real thing, you must run the Help Compiler and compile the whole database, which can take a long time. Inserting the correct footnote tokens with the correct footnote values is a tedious and error-prone process. Remember that you are using the features of a word processor to create links and chains in a textual database, a text editor to associate the link identifiers with numbers in the project file, and a software-development environment to put the numbers in the source code. There are no built-in integrity checks. Nothing ensures that you properly coordinate the contexts and topic identifiers among the three files. Some, but not all, of these problems go away when you use a help-authoring tool.

Microsoft Windows Help Author

The Microsoft Developer Network CD contains an "Unsupported Tools and Utilities" section that includes a program named "Help Author," which is easy to use and well documented. The MSDN CD-ROM includes as a bonus an extensive Help Authoring Guide that covers the creative side of the job.

Help Author has two parts: an application named "Help Project Editor" and a Word for Windows template. The Help Project Editor uses dialog boxes to collect the information for the ASCII project file. You don't have to deal with that file again. It also automates the interface with Word, launching it with the template installed and the RTF file loaded.

Answers For Sale

Match the questions below to the books that answer them:

1. How can I exchange files and email with colleagues in Japan?

2. I can design software, but how do I make a business out of it?

3. How can I find out about all the TeX tools I'll need for my SGML environment?

4. How can I zero in on the array enhancements in Fortran 90?

5. How do I let multiple developers change files and perform builds on the same product?

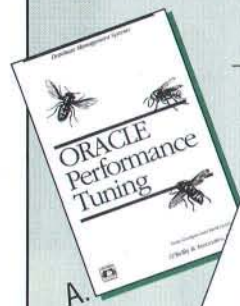
6. Our Oracle application ran fine in testing, but the response is terrible now that we're in production. What can I do?

7. How do I set up configuration files to handle cross-environment applications?

If you need UNIX answers, call for our catalog.

O'Reilly
& ASSOCIATES, INC.

800/998-9938 • fax 707/829-0104 • order@ora.com



A.



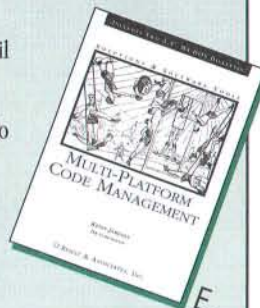
B.



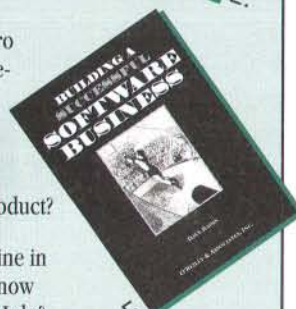
C.



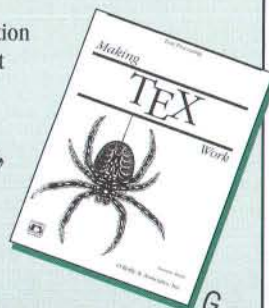
D.



E.



F.



G.

CIRCLE NO. 887 ON READER SERVICE CARD

A help database can contain more than one RTF file, and Help Project Editor keeps them in a list. You can also launch the Help Compiler and WINHELP to view the currently compiled database.

The Word template adds three tool buttons to the Word tool bar. The first one opens a dialog box that lets you change the footnote values in the current topic. You can add, change, and delete the topic's title, context mnemonic string, keywords, browse sequence, and so on, all without having to deal with Word's footnote commands. The second tool button opens a dialog box that lets you insert jump and pop-up links into the database, automatically applying the underlined and hidden text attributes. If you select a phrase that is already in the text, the operation uses it. Otherwise it inserts whatever you put in the dialog box as the link phrase.

The third tool button compiles only the current topic into a temporary help database and calls WINHELP to display it. You preview a topic—text, graphics, sound, and movies—exactly as the user sees it and without recompiling the entire database. What you see in Word and what WINHELP displays is usually quite different. You need to view your progress in small increments, and this feature supports that need. It is Help Author's strongest advantage over the other tools, and anyone who develops a sizable help database wants this capability. The other tools do not have it.

Help Author does not integrate graphics and multimedia tools. You still have to launch them yourself and write their files into the proper subdirectory so that the Help Compiler finds them.

An amusing side to Help Author is that its own help database has context errors. Most of the Help buttons on its dialog boxes link to help topics that do not exist, although there are topics in the database to cover the functions of the dialog boxes. Nonetheless, Help Author smooths several of the wrinkles out of the manual procedure, automates most of the tedium of using Word to build the database, and is well worth trying. As a Windows developer, you should have the MSDN CD-ROM, anyway. Help Author is a bonus.

Windows Help Magician

Windows Help Magician from Software Interphase has some good features, some bugs, and some annoying quirks. Among its quirks is that the setup includes a package called "Bitmap Magician." Its purpose is to let you build a pseudofont by converting the characters in an existing font into bitmaps that you can include in the help database. Help Compiler uses only a few fonts

and doesn't accept all of the characters in the fonts it does allow. For example, you can put the trademark character in the text, but the Help Compiler deletes it from the help database. Bitmap Magician solves that problem.

When you run it, Bitmap Magician asks you to select a font. When you do, it says that there was an "overflow" with no explanation of what that means. Next, you learn that you are looking at only a demo version. The dialog advis-

Because RoboHelp is a Word template, you can see its source code by opening the macros

es you how to order the real thing. When you acknowledge that piece of good news, the program changes the mouse cursor to an hourglass and leaves it that way. Most Windows users would think that the system is hung up. Not really. You can use the hourglass cursor as if it were an arrow. Close the program, delete its icons from the Program Manager group and proceed to the Help Magician itself.

The second annoyance is the overall appearance of Help Magician's windows. The design is an example of a designer gone wild with enthusiasm over 3-D sculptured controls but without the design skills to know how to use them. Everything in the application window and all of the menus and dialog boxes are broadly sculptured. I understand that this is a matter of taste, but I have never seen anything quite like this. There is no menu bar, only a big, fat, sculptured tool bar. When you punch it, it pulls down a menu, also sculptured. The menus use tool buttons. There are the usual File, Edit, Options, Help, and other menus, but they are all represented by ugly tool buttons. The real tool bar is sculpted at the bottom of the window. The overall appearance detracts from the program's functionality.

Help Magician works with its own database format while you compose the help information. Then it converts to RTF format to run the Help Compiler. It launches the word processor of your choice but does not provide templates.

Help Magician has its own editor. That's a good idea, but it's not particularly well implemented. Text that you select for titles and links is surrounded by vertical bars. The vertical bars come in pairs and have to be balanced. You

cannot distinguish a starting vertical bar from its terminating vertical bar in a pair. You cannot distinguish two sets of different pairs of vertical bars. A help topic with centered or justified text, a title, and some jump links displays with a mélange of pairs of vertical bars. It's hard to read.

I went through the tutorial process and then tried to add a MIDI sound bite to the tutorial's help document. Somehow I messed up the database. Somehow the MIDI insertion upset the balance of the vertical bars in the line of text. I could not build the RTF file or delete the line. Help Magician stubbornly issued error messages no matter what I tried. Finally, I deleted the entire topic, resulting in lost work.

Next I moved to my own project and imported the RTF file that I built using the manual procedure and Help Author. Without making any changes, I tried to rebuild the Help Magician database into a new RTF file. Help Magician reported another unbalanced marker, this time telling me that I could delete it with a Ctrl+bracket key combination. It didn't tell me that before. I don't know where the unbalanced marker came from. I looked at the original RTF file, and everything looked okay. I deleted the unbalanced marker and saved the RTF file.

Help Magician uses a single-font edit box, and your view of the help text is completely unlike what it is going to look like in a help window, far more so than if you are using Word. Centered text is not centered, and margins are not shown. Those distracting vertical bars are everywhere. To move from topic to topic, you have to change the page number in an edit box at the bottom of the screen and press the Enter key.

There is no preview mode. You can test the database, which displays the help in the same single-font, vertical-bar format and lets you exercise the jumps and popups. However, to see the real thing, you must compile the entire database and run WINHELP.

In one place, the RTF import mangled a graphic token. You could see where some of the RTF protocols were exposed as if they were text. The result was that the Help Compiler could not find the bitmap file. I was able to fix the token in the editor, but the graphic lost its text-centered attribute. I found no way in Help Magician to center or otherwise justify text or tokens. Similarly, there seems to be no way to set margins other than to launch Word, do it from there, and import the RTF file again. Not a good idea, given the import mangling. There were several other places where the import mangled the RTF file. I had to fix them in the Help Magician editor,

FAST, PORTABLE C DEVELOPMENT TOOLS

- ✓ Client/Server
- ✓ Proven Performers
- ✓ Develop Royalty-free with c-tree

c-tree:
your best solution
for database needs
on Windows
and OS/2!

✓ Unsurpassed Portability

✓ Source Code

✓ Customizable

✓ Powerful Report Generation

More Capable

More Affordable

Expensive DBMS Servers

Lightweight Solutions lack features and flexibility you may need to accomplish your mission.

Heavyweight Solutions will cost a small fortune before you even leave the ground.

FairCom Offers A Full Line Of High-Performance Servers & Development Tools At Reasonable Prices.

FairCom Servers SQL & non-SQL

These multi-threaded database servers offer a seldom-found solution for developers who demand control. While one may mandate SQL access, another's real-time demands may not tolerate the overhead associated with SQL. Our unique servers offer the full range of data accessibility: low-level speed; convenient ISAM-level; compatible SQL-level.

Complete **data integrity** is achieved with multi-user transaction processing. Recovery of all committed transactions after a failure is fully automatic. The C developers 'CLIENT/SERVER' solution: **DOS nodes to UNIX servers**; full commit and rollback; roll forward; precise control over your data and/or data base model; large files (4GB); complete client source code.

d-tree Toolbox Application Development

Productivity tools with: a complete **portable screen handler**; data dictionary; code generation; easy to use data base interface; menus; help text; data validation. d-tree's dynamics allow **runtime control** of program resources (screens, files, edits, etc.) not found in any other development package. Resources can be changed in memory, and/or swapped on/off of disk at runtime. If you do application development on multiple platforms (DOS/UNIX), or you're debating 4GL versus C development, d-tree is for you.

c-tree Plus High Performance Data Management

Based on the most advanced B+ Tree routines available today, c-tree Plus gives you unprecedented control over your file management needs. With unparalleled sophistication, c-tree Plus has established itself as the premier choice for commercial development. Use the low-level routines or take advantage of the high-level ISAM routines for high speed random or sequential access. c-tree Plus is distributed in **complete C source code** and is known for its portability and **royalty-free** licensing policy. Whether for single-user, multi-user or client-side application development, c-tree Plus delivers. Transaction processing is included in c-tree Plus- call for a complete list of features.

Natural Query Natural Language Tool

Produces ad hoc reports quickly and easily from English sentences. It is the first low-cost, natural language tool designed for developers. The QBE option has easy-to-use, pick and choose menus, making report specifications a snap.

CIRCLE NO. 128 ON READER SERVICE CARD
Call today for more information:



(800) 234-8180

FAIRCOM[®]
corporation

4006 W. Broadway · Columbia, MO 65203
(314) 445-6833 Fax (314) 445-9698

FAIRCOM EUROPE
Via Sottocorna 15/17 · ALBINO (BG) ITALY

and, once again, had no way to set the margins or control the justification.

I launched Word from Help Magician to look at the saved RTF file. It was different now. All of the link phrases and their context identifiers were displayed with a strike-through font and Help Magician had added a bunch of its own footnotes. Even though I had a copy of Word running, Help Magician launched a new copy. (Help Author's launch was smart enough to use the copy of Word that was already running.) The strike-throughs and new footnotes didn't seem to have hurt anything.

Help Magician launches the Help Compiler, WINHELP, Word, HotShot Editor, Paintbrush, and the Sound Recorder. It installs Microsoft Video playback software and shows you how to add audio, video, animation, and MIDI to a help database. Before you use Help Magician on a real project, however, spend some time with its tutorial and get a feel for how it works and where the bugs are. You might like it, and you might not.

RoboHelp

RoboHelp from Blue Sky runs on top of the word processor. The current release supports Word for Windows only, although Blue Sky plans support for other packages. It won't be an easy port because the main part of RoboHelp is implemented as a Word document template with macros written in the Word-Basic programming language. There are some other executable utilities, including one that launches a RoboHelp project, but you can just as easily launch it yourself from Word simply by opening a document that includes the RoboHelp template. This implementation is a dramatic example of what a programmer can do with WordBasic.

The RoboHelp template modifies Word's menus and tool bar and adds a floating tool bar that stays on top and to the right of the document while you are editing. The commands open dialog boxes to establish and change the characteristics of the help project. You add topics, jumps, popups, graphics, search keywords, topic titles, and context mnemonics by pressing tool buttons and filling in the dialogs. RoboHelp manages the document and the help project file and does not use its own database format; it uses the Word document format. One tool button writes the RTF file from the Word document. Another runs the Help Compiler to build the help database.

You can preview a topic by pressing a button, but the preview is not much better than what Word is already showing you. For example, it doesn't show graphics. In fact, RoboHelp's topic preview is

not as good as viewing the topic in Word. The preview justifies all of the text in the left margin regardless of how you have the margins and paragraphs set up. It shows the graphics-insert tokens just as they appear in the document. It's a feature they could have left out.

There are other things that I would change. When you open a RoboHelp document, its Word template defaults the Edit/Find command to search for hidden text, presumably so you can find the jump and pop-up links. Most of my searches are for text in the document, which is not hidden, so I have to change the Find property every time. The "H" tool button that the template added changes selected text to unhidden, an odd choice for this button. I wouldn't think you'd need this one very often. A better choice would be to toggle hidden text into and out of view. Some of the time you want to see the links; other times you want the text to line up more like it does when WINHELP displays it.

Like Help Magician and unlike Help Author, RoboHelp does not use an existing running copy of Word; it launches its own. Because RoboHelp is a Word template, you can see its source code by opening the macros. Furthermore, if you don't like the behavior that I just described or anything else, you can modify it by changing the WordBasic code. You could even add your own adaptation of Help Author's indispensable topic-preview feature.

RoboHelp builds context-identifier files you can include in C++ and Visual Basic programs. It includes a VBX control that adds a help button to a dialog box and prompts you for the associated context identifier. A Screen Capture utility manipulates pictures from the Clipboard. It runs in the background waiting for you to put some graphics in the clipboard. When you do, it pops up with an image-processing tool that lets you modify what you captured into a bitmap for your help database. There is also an icon-composition tool included with RoboHelp.

Summary

It's not hard to pick a favorite from these alternatives. The manual approach worked, but was tedious. I prefer it over Help Magician, however, which seems to be not quite ready for prime time. Help Author is an order of magnitude better than the manual setup, and RoboHelp is far and away the best tool for the job that I've seen.

When I started this project, I went looking for "Visual Help." Although I didn't find exactly that, I am satisfied that there are tools that make the job easier. I do think, however, that a need for such a product exists. It would have

all of the best features of the three packages discussed here. In addition, its editor would emulate the WINHELP display—that's the "visual" part. The tool would use RTF as its native database format and could emulate the jumps, popups, and multimedia features of WINHELP. Unlike Word, it would display graphics without taking all day. Such a program would eliminate Help Compiler until the end of the help document development project. This would be a valuable product. If you build it, they will come.

DDJ

To vote for your favorite article, circle inquiry no. 9.

For More Information

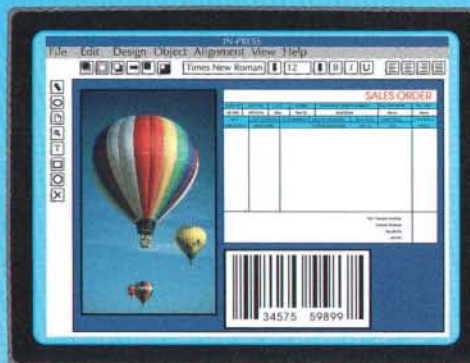
Microsoft Developer Network CD
Microsoft Corp.
One Microsoft Way
Redmond, WA 98052-6399
800-759-5474

The Windows Help Magician
Software Interphase Inc.
82 Cucumber Hill Rd., #113
Foster, RI 02825
401-397-2340

RoboHelp
Blue Sky Software
7486 La Jolla Blvd., Suite 3
La Jolla, CA 92037-9582
800-677-4946

FORM = FUNCTION

What you see is no longer only what you get.
With IN-PRESS,™ you get a report writer, a form editor
and programming language source code, too!



Languages

C/C++
CA-Clipper 5.01 or above
CA-dBase
CA-Realizer 2.0
Clarion
FoxPro 2.0/2.5 for DOS
FoxPro 2.5 for Windows
Force
Paradox for Windows
Pascal
Visual Basic for Windows
Visual Basic for DOS
Generic Function Calls

Visually create reports and forms:

- fully banded reports tied to data
- graphics (.bmp, .pcx and .tiff)
- bar codes and labels
- invoices and other business forms
- multi-fonted text

Automatically generate source code:

- in most popular computer languages
- portable between languages
- fully customizable
- easy-to-use functions
- royalty free

Take advantage of Windows GDI compatibility to work
with any printer, or make your reports really fly with
direct PCL and PostScript support in Windows and DOS.

So have your cake and eat it. Get control with ease of use.

Here's a tool that not only looks good, but also speaks your language.

Try IN-PRESS risk free with a 30 day money back guarantee.

For literature or to order IN-PRESS, call 1-800-533-3183.

In PA: (215) 692-8130 • FAX: (215) 692-8172 • BBS: (215) 631-5724

D A B I W A
LTD.

CIRCLE NO. 712 ON READER SERVICE CARD



Algorithms for Directed Graphs

A unique approach using genetic algorithms

Salvatore R. Mangano

Directed graphs underlie any tool that displays a tree diagram, class-relationship diagram, or entity-relationship diagram. As such, you might expect a CASE tool to provide an optimized directed-graph drawer. However, most CASE tools I'm familiar with punt when addressing this problem. Although an algorithm for drawing a directed graph like that shown in Figure 1 is straightforward, a general-purpose graph drawer that draws graphs in an aesthetically pleasing format is difficult to create and computationally expensive. So, CASE tools usually use a few simple rules to get an initial layout and then allow the user to clean things up by dragging objects around. Putting the burden of "pretty drawing" on the user wastes time better spent on the design.

This article looks at a novel solution to this problem using the emerging technology of genetic algorithms (GAs). Specifically, I'll use EOS, my company's C++ GA application framework, and Microsoft's Visual C++ to develop a module for optimizing the aesthetic layout of directed graphs. I'll create a Windows-hosted test application that exercises this module on randomly created graphs. The intent of this article is not to produce a commercial-grade graph drawer, but rather to demonstrate the use of GA technology on a nontrivial and unique problem. Since most programmers' first exposure to GAs is usually on a function-optimization problem, this article provides some insights on the advanced use of GA techniques.

The Technique

A GA is an algorithm that works with a population of potential solutions to a problem. Through means analogous to

biological natural selection, it evolves better and better solutions to the problem. To accomplish this, the user of a GA must first find a way to encode the problem into a string of bits. The bits are analogous to genes and the strings to chromosomes. The encoding of a solution as bit strings is often called the "genotype" and the decoded solution, the "phenotype." The genotype is mapped to the phenotype by the decoding function. The next step after the encoding is the measure of fitness. Fitness is one of the core elements that appear in every variation of a GA. Calculation of fitness involves mapping a solution onto a positive number such that greater numbers correspond to better solutions. This mapping is accomplished by the fitness or objective function.

The second core feature of every genetic algorithm is a population of individuals. At any time during the execution of a genetic algorithm, there exists a population of candidate solutions to the problem (individuals consisting of a genotype and a phenotype). The initial population is usually generated randomly. The process of transforming this initial, mediocre population into a population containing near-optimal solutions is the heart of the GA. It proceeds by iterations of the following genetic operators: selection, reproduction, crossover, and mutation.

Selection is the process by which candidate individuals from the current generation are chosen to produce the next generation. Selection is a survival-of-the-fittest strategy. After two individuals are selected, a weighted coin is flipped to determine if the individuals will mate to produce two new offspring or simply be placed in the next generation as is. Mating is accomplished by the crossover operation. The probability of mating is called the "crossover probability" (pc). The simplest form of crossover, called "single point," is shown

in Figure 2. As each bit is copied from parent to child, it is subject to mutation based on the mutation probability (pm). Pm is usually very small, relative to pc. Iterations of selection, reproduction, and mating are repeated until a new population is created. At this point, the fitness values of the new population are recalculated, and the process repeats until either some acceptable solution is found or an upper time limit has been reached. The GA described above can be summarized by the procedure shown in Figure 3.

The Tools

A GA framework is useful due to the large number of GA variants that can be produced by altering one or many of the steps in the basic algorithm. GA re-

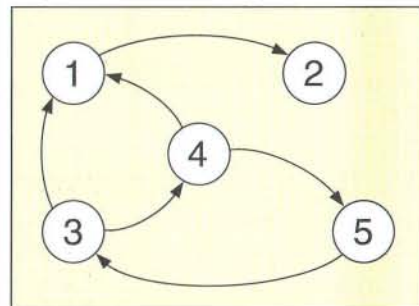


Figure 1: Typical directed graph.

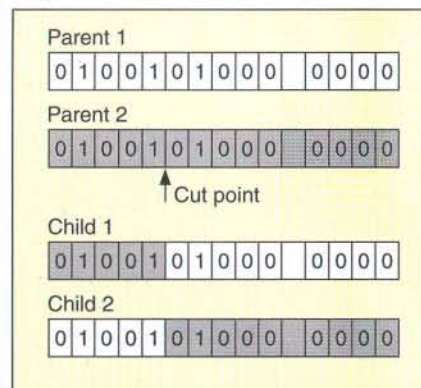


Figure 2: Single-point crossover.

Sal is president of Man Machine Interfaces. He can be reached at 555 Broad Hollow Road, Suite 230, Melville, NY 11747 or on CompuServe at 72053,2032.

 **FOCUS**

GENEXUS™

SYNON

SPEEDWARE®

ADELIA

LAOSA

 **BLYTH SOFTWARE**

IBM

PROGRESS
SOFTWARE

JDEdwards®

GUPTA


TRACK


JBA
People and Products in Software

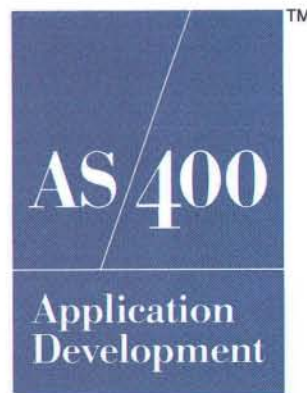
AS/SET™

X-ANALYSIS™

What they all have in common: **Quality**

As partners in the IBM AS/400 Application Development Program, they've earned the right to display the emblem below. For more information on these products and vendors, call 800/365-4426, extension 260.

International fax:
1 507/253-3131.



All trademarks are trademarks and registered trademarks of their respective owners.


```

Initialize a random population and measure its fitness.
WHILE (stopping criteria has not been reached)
BEGIN
  WHILE (next generation is not full)
  BEGIN
    Select 2 parents randomly based on fitness.
    IF (Flip(pc)) THEN
      Cross parents (mutating with probability pm)
      and place children in next generation.
    ELSE
      Place parents into next generation untouched.
  END
END
Solution with highest fitness is the answer.
    
```

Figure 3: Pseudocode detailing the GA process.

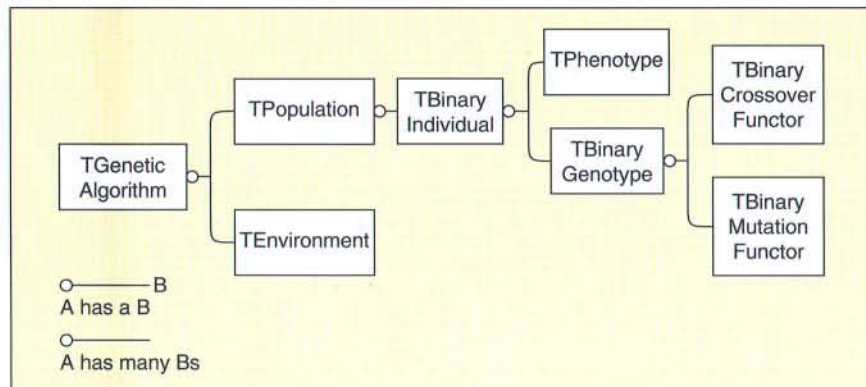


Figure 4: Relationship between classes.

(continued from page 92)

searchers have invented several variations on selection, reproduction, crossover, and mutation. Each variation can be mixed and matched to produce a unique GA variant. Object orientation turns out to be an ideal technique for expressing these variations. Through an adept combination of inheritance and composition, all the GA variants can be expressed. This ultimately allows you to code a GA using the basic technique and then try variations by instantiating different classes.

Although EOS consists of over 80 classes, I'll restrict this discussion to a small relevant subset—*TBasicGA*, *TPopulation*, *TEnvironment*, *TBinaryIndividual*, *TGenotype*, *TPhenotype*, *TBinaryCrossoverFunctor*, and *TBinaryMutationFunctor*. These bases consist of many derived classes that implement variants of the basic GA. Other classes exist to implement special-purpose features. Each of the classes listed encapsulates a different behavior of the overall GA. *TBasicGA* is the genetic-algorithm interface class. *TPopulation* is a collection class that holds instances of *TIndividual*. *TEnvironment* encapsulates the GA's parameters—pc, pm, the random-number generator, and other statistical information. *TBinaryIndi-*

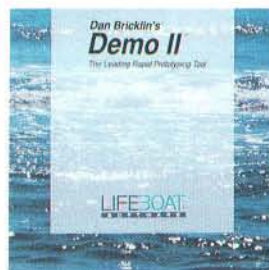
Dan Bricklin's® Demo II™

The Leading Rapid Prototyping Tool

Turn raw ideas into visual concepts

Experience for yourself why over 40,000 people have made Demo II version 4.0 the leading tool for producing program prototypes, demonstrations and tutorials.

- Demonstrate commercial software to potential customers without shipping live software.
- Produce effective tutorials that interactively teach products.
- Create Computer Based Training for a fraction of the cost of dedicated CBT authoring software.



Version 4.0 Includes:

- Super VGA Support
- Full and Partial Screen PCX Imaging
- Animation
- Bitmapped Font Overlays
- Built-In Mouse Support
- Autosave Option

Suggested Retail Price \$249

LIFEBOAT®
SOFTWARE
800-447-1955 **908-389-0037**

1163 Shrewsbury Avenue • Shrewsbury • NJ • 07702

CIRCLE NO. 921 ON READER SERVICE CARD

Demo II is a trademark of Lifeboat Software. Dan Bricklin's is a registered trademark of Daniel Bricklin.

vidual is an interface class that unifies instances of *TGenotype* and *TPhenotype* into a single object. *TBinaryGenotype* encapsulates the binary genetic coding of the problem as strings, and it provides an interface to the crossover and mutation classes. *TPhenotype* encapsulates the decoding function and the fitness function. It is the main class from which you derive to build a GA-based application. *TBinaryCrossoverFuncutor* and *TBinaryMutationFuncutor* are classes that encapsulate the operations of crossover and mutation. These classes are called functors because they are functional objects. Functors are used so various flavors of crossover and mutation can be plugged in or out of a genotype without recoding any of the genotype's methods. Figure 4 shows the relationship between these classes,

The GA Module

To derive a genetic encoding, I'll formalize the problem we are attempting to solve. We are given some arbitrary directed graph, as well as a grid where each cell represents a potential home for the graphical depiction of a node in the graph. The goal is to find an assignment of nodes to cells such that when the arcs are drawn, we get an aesthetically pleasing picture. Stating the problem in this way makes some crucial assumptions that may not be true in a real situation. First, I assume that the nodes, when drawn, are of equal size. Second, I assume that once I have assigned nodes to cells, the arcs can easily be drawn to complete the best possible drawing. (In other words, we need not optimize the drawing of arcs.) Third, I assume that the nodes are equally spaced in a grid and not arbitrarily placed on the output screen. I make these assumptions to simplify the example and the code. A more general solution is certainly possible using GAs.

Genetic Encoding

If each node in the graph is assigned a sequential number, then the problem can be viewed as a mapping of each node number onto an (x, y) coordinate in the grid. The mapping that keeps connected nodes close together and produces the fewest arc crossings will yield more aesthetically pleasing drawings. Other domain-dependent criteria may come into play when determining better drawings, but I ignore this possibility to expedite the solution.

Given the above formalism, the encoding treats the bit string as a series of (x, y) pairs. The first pair assigns node0 to grid cell (x0, y0). The second assigns node1 to (x1, y1), and so on. This encoding allows collisions (two or more

nodes are assigned to a single cell), so I need a collision-resolution procedure. A problem like this often arises in GAs, and there are several approaches to handling it. Some programmers assign very low fitness values to illegal genotypes. Others attempt to repair illegal genotypes before decoding them. Still others create special-purpose crossover and mutation operators that do not allow illegal genotypes to arise in the first place. In this case, I'll resolve a collision by searching for the closest empty cell to the one assigned, according to a fixed procedure. This is similar to a repair technique, but we are repairing the phenotype instead of the genotype.

Given that I have a graph with N nodes and a grid that is X cells wide and Y cells high, I can calculate the required length of the bit string using the equation in Figure 5. If X and Y are not powers of 2, then it is possible that (x, y) pairs can be encoded such that either x or y is greater than X or Y. I resolve this problem by always decoding x modulo X and y modulo Y. The decoding is implemented by a *TPhenotype* derived class called *CGraphDrawingPheno*. This class

$$\text{Chromosome Length} = N * (\lceil \log_2(X) \rceil + \lceil \log_2(Y) \rceil)$$

Figure 5: Equation to calculate the required length of a bit string.

A Total Forms Processing Tool Kit

AutoData SDK™ II

Windows 3.1 DLL

- Hand-Printed
- Mark Sense
- OCR
- Bar Code Recognition

Empower Your Software To Read

The AutoData SDK II Software Developers Kit provides state-of-the-art recognition and imaging tools to automate data entry from scanned or faxed forms.

A Unit of Electro-Sensors, Inc.

AutoData Systems

10365 West 70th Street • Eden Prairie, MN 55344-3446
Phone: 612/941-8180 • FAX: 612/941-7312
Toll Free: 800/662-2192

OPTLINK® IS NOW ROYALTY FREE!

OPTLINK® 5.1 for Windows

Shrink your program file sizes another 50%!

Enables Windows developers to generate compressed self-loading executables (.EXE) and Dynamic Link Libraries (.DLL), thus reducing file sizes an additional 50%.

Benefits include:

- Loads applications faster
- Saves you \$'s - fewer diskettes to ship
- Speeds up installation time
- Complicates reverse engineering
- Reduces your customers' disk requirements
- Only requires relinking

Other reasons to use OPTLINK:

- Fastest Windows linking
- Unrivalled capacity
- Builds the smallest programs
- No limits on debug information
- Eliminates RC, IMPLIB, IMPDEF



OPTLINK v5.1 for Windows also adds support for Borland C++ v4.0, Microsoft Visual C++ v1.5 and Symantec C++ v6.1. Full debug support for CodeView and Turbo Debugger. 30 Day Money Back Guarantee. Price: \$350.



SLR Systems, Inc.
1622 N. Main Street, Butler, PA 16001 USA
Phone: (412) 282-0864; Fax: (412) 282-7965

CIRCLE NO. 216 ON READER SERVICE CARD

THERE IS NO SUCH THING AS A FREE MAKE!

Why pay for a Make utility when one comes free with your compiler? Because the moment you start using it you start paying for it. You pay with your time, effort and frustration in trying to keep your software projects up-to-date.

Opus Make excels where your Make falters. For large projects and interfacing to version control systems, for transparent handling of your Microsoft and Borland makefiles, for PolyMake makefile processing at twice the speed for half the price, for exceptional technical support at no additional charge, Opus Make version 6.0 is the only economical choice.

Your time isn't free. Neither is your sanity. Find out why Andrew Binstock says "Opus Make is the most compelling reason for not using the make utilities bundled with today's compilers."

Opus Make
VERSION 6.0



Opus Make version 6.0 features include:

Makefile Compatibility: Processes Intersolv PolyMake 4.0™, Microsoft™ NMake and PWB, and Borland Make™ makefiles (even auto dependencies).

PolyMake Emulation: Emulation of PolyMake v4.0!

Object Library Maintenance: Maintain libraries without object file clutter - thus saving disk space.

Version Control Support: Access source files in One Tree SourceSafe™, Burton Systems TLIB™, Intersolv PVCS™, and MKS RCS™ version control systems.

Other features: Multiple-directory support • Memory swapping • Pattern-based inference rules • Multiple targets created from single source • Automatic and in-line response files • Queued shell lines • Conditional, looping and include directives.

Opus MKMF V3.2 included: MKMF is our makefile and dependency generator. It quickly builds and maintains your makefiles with a minimum of input. It understands C-preprocessor directives and resource compiler files. If you hate building makefiles by hand this is the tool for you!

OPUS Software, Inc. • 1032 Irving Street, Suite 439 • San Francisco, CA 94122 • USA
Phone: (415)-664-7901 • Fax: (415)-664-5624
For More Information Call 1-800-248-OPUS ext 44

Both DOS
and OS/2
for only
\$165.00

Opus Make and OPUS MKMF are trademarks of OPUS Software, Inc. All other trademarks are owned by their respective companies.

CIRCLE NO. 599 ON READER SERVICE CARD

PROGRAMMER'S WORKBENCH

contains a two-dimensional matrix that will represent the grid. The genotype will be decoded so that each entry in the matrix will receive the node number of the node assigned to that grid position. Empty positions will be assigned

A GA framework is useful due to the large number of GA variants that can be produced

a value of 0. The decoding of the genotype is implemented by *CGraphDrawingPheno::Decode()* member function; see Listings One and Two, page 106. This function uses a reference to a graph drive class to determine the number of bits in each component in the encoding. It copies these bits to buffers to be converted to integers by the utility functions *AllelesToInt()*. "Allele," a term borrowed from biology, refers to the value expressed by a gene. Once the node, its row, and column in the grid are decoded, the member function *GetNearestEmptyCell()* is called to resolve the possibility of a node already existing at the desired location.

The Fitness Function

Now that I have a way to encode the placement of a graph's nodes on a grid, I need a technique for evaluating each placement's fitness. There are many ways to do this, depending on what you consider to be an aesthetically pleasing layout. When deriving this fitness function, I let intuition guide me in the initial derivation and then experiment to tweak the function so it works well for a variety of graphs. The function I ultimately arrived at can be seen in the *CGraphDrawingPheno* class's *CalcFitness()* member function; see Listing Two. The idea behind this function is to reward genotypes that decode into drawings where nodes connected by an arc are adjacent or close and to penalize when nodes are adjacent but not connected. This is done on a node-by-node basis so the resulting fitness value is a measure of how well nodes of the entire graph were assigned to grid locations. Notice that I completely ignore arc drawing for simplicity. The remaining members of *CGraphDrawingPheno* implement construction, destruction and copying. I also include some private-utility functions that encapsulate the testing

for adjacency and the calculation of distance between grid cells. These can be used in experimenting with variations of the fitness function.

I include three other classes in this module: *CGAGraphDriver*, *CGraphDrawerGA*, and *CWordMatrix*. *CGAGraphDriver* is an interface class that collects information (such as number of nodes in the graph and the size of the grid) before the GA and its associated objects are initialized; see Listings Three and Four (page 107). A very important function in this class is *CalcChromosomeLength()*, which, based on the number of nodes and the size of the grid, determines the number of bits necessary in a chromosome to encode the problem. Also included in this class are functions for drawing the optimized and unoptimized views of the graph. These use Windows-specific functions, but the logic can be easily ported to other graphics systems.

CGraphDrawerGA is derived from the EOS class *TBasicGA*. It overrides the population-creation function and several reporting functions useful for testing the GA's performance before it is embedded into a larger application. Listings Five and Six (page 147) show the class declaration and implementation of *CGraphDrawerGA*. The important function here is *CreatePopulation()*. This function determines the characteristics of the genotype, phenotype, and the population. I use a two-point crossover operator genotype (instead of single point) because this tends to work better with longer chromosomes. I also use a population technique known as *Elitism*, which ensures that a certain number (in our case, two) of the best individuals from the previous generation make it to the next generation. This improves performance on some types of problems.

The utility class *CWordMatrix* implements a 2-D matrix of WORDS. Its implementation makes use of the MFC's *COBArray* and *CWordArray* to create an array of *CWordArrays*.

The Test Program

To test the graph-optimizing GA, I created a simple Windows-hosted application using Visual C++. I used AppStudio, AppWizard, and ClassWizard to automate the creation of this program. The program uses two dialog boxes. The first dialog box allows me to specify the graph logically in terms of the number of nodes and their connections. The second allows me to specify the bounds of the grid and trigger the GA optimization of the graph on the grid. I also include options for displaying the optimized and unoptimized views of the graph. The

optimized view is the best solution found by the GA; the unoptimized view is an arbitrary drawing of the graph from first node to the last. Due to the length of the test program, the complete listings for this project are available electronically; see "Availability," page 3.

Conclusion

Experiments that I have conducted using the GA-based graph-drawing module demonstrate that GAs present a viable solution to the problem. By improving the fitness function and by possible use of custom genetic operators, I believe that this technique will also work in commercial CASE tools.

References

Coplien, James O. *Advanced C++: Programming Styles and Idioms*. Reading, MA: Addison-Wesley, 1992.

Goldberg, David E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

Michalewicz, Zbigniew. *Genetic Algorithms + Data Structures = Evolution Programs*. New York, NY: Springer-Verlag, 1992.

DDJ

(Listings begin on page 106.)

To vote for your favorite article, circle inquiry no. 10.

WHEN THE DEBUGGING GETS TOUGH... REACH FOR ONE OF PERISCOPE'S "BUTT-SAVING" DEBUGGERS.



"Great product, has already saved my butt more than once," says one relieved Periscope user.

Just push the "panic button" on the break-out switch (comes with all Periscopes) when your system locks up.

SOFTWARE-ONLY DEBUGGERS for DOS, Windows and OS/2...

- Use Periscope/EM to debug real-mode DOS software, including applications, drivers, TSRs, and interrupts. Run Periscope/EM from conventional DOS memory or extended memory.
- Use Periscope/32 to debug system-level software running under Windows 3.x, OS/2 2.x, or your own 32-bit operating environment.

RENT OR BUY THE HARDWARE when you need the power of an ICE...

- The Periscope Model IV hardware adds the power of an ICE to both Periscope/EM and Periscope/32. Debug hardware interrupts, communications software, real-time software, and in any situation where you need zero slowdown, no-impact tracing or monitoring.

"I've checked out all the various other debuggers as they've clamored for my keyboard, but Periscope is the one I've stayed with."

Steve Gibson, *InfoWorld*, May 24, 1993

Call us TOLL-FREE 800/722-7006 to order
or for product details and prices.

PERISCOPE®

1475 PEACHTREE ST., SUITE 100 • ATLANTA, GA 30309 USA
404/888-5335 • FAX 404/888-5520 • 800/722-7006 (US & CANADA)

CIRCLE NO. 882 ON READER SERVICE CARD

Listing One (Text begins on page 38.)

```

/* Blowfish.h */

#define MAXKEYBYTES 56          /* 448 bits */

short opensubkeyfile(void);
unsigned long F(unsigned long x);
void Blowfish_encipher(unsigned long *xl, unsigned long *xr);
void Blowfish_decipher(unsigned long *xl, unsigned long *xr);
short InitializeBlowfish(char key[], short keybytes);

```

Listing Two

```

/* Blowfish.c */

#include <dos.h>
#include <graphics.h>
#include <io.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <alloc.h>
#include <ctype.h>
#include <dir.h>
#include <bios.h>
#include <Types.h>

#define N 16
#define noErr 0
#define DATAERROR -1
#define KEYBYTES 8
#define subkeyfilename "Blowfish.dat"

static unsigned long P[18];
static unsigned long S[4,256];
static FILE* SubkeyFile;

short opensubkeyfile(void) /* read only */
{
    short error;
    error = noErr;
    if((SubkeyFile = fopen(subkeyfilename,"rb")) == NULL) {
        error = DATAERROR;
    }
    return error;
}

unsigned long F(unsigned long x)
{
    unsigned short a;
    unsigned short b;

```

```

    unsigned short c;
    unsigned short d;
    unsigned long y;

    d = x & 0x00FF;
    x >>= 8;
    c = x & 0x00FF;
    x >>= 8;
    b = x & 0x00FF;
    x >>= 8;
    a = x & 0x00FF;

    y = ((S[0, a] + (S[1, b] % 32)) ^ S[2, c]) + (S[3, d] % 32);
    /* Note: There is a good chance that the following line will execute faster */
    /* y = ((S[0,a] + (S[1, b] & 0x001F) ^ S[2, c]) + (S[3,d] & 0x001F); */
    return y;
}

void Blowfish_encipher(unsigned long *xl, unsigned long *xr)
{
    unsigned long Xl;
    unsigned long Xr;
    unsigned long temp;
    short i;

    Xl = *xl;
    Xr = *xr;
    for (i = 0; i < N; ++i) {
        Xl = Xl ^ P[i];
        Xr = F(Xl) ^ Xr;

        temp = Xl;
        Xl = Xr;
        Xr = temp;
    }
    temp = Xl;
    Xl = Xr;
    Xr = temp;

    Xr = Xr ^ P[N];
    Xl = Xl ^ P[N + 1];

    *xl = Xl;
    *xr = Xr;
}

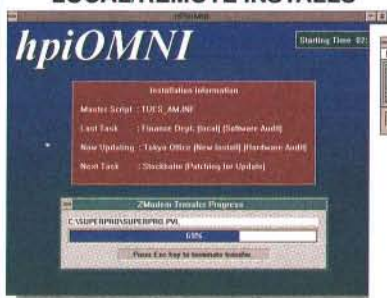
void Blowfish_decipher(unsigned long *xl, unsigned long *xr)
{
    unsigned long Xl;
    unsigned long Xr;
    unsigned long temp;
    short i;

    Xl = *xl;
    Xr = *xr;

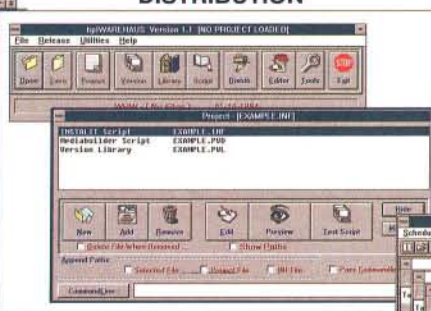
```

Let US install any of our products on your computer within the hour!¹ Plus...on request we could send you our product line²... 24 hour vending. How'd we do that? With HPI technology. And you can too...with hpiOMNI/Crypto.

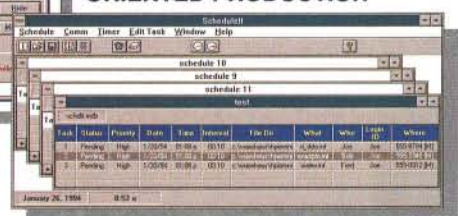
LOCAL/REMOTE INSTALLS



DISTRIBUTION



EXTENSIBLE PROJECT ORIENTED PRODUCTION



- no hidden royalties or relicensing
- the most comprehensive and advanced distribution systems available at any price, guaranteed!
- network, ZModem, or diskette distribution
- free trial / 90 day money back guarantee
- INSTALIT language or Visual Basic Custom Controls
- scripted / non-scripted
- source available
- Extended Edition includes patching, duplication, and the revolutionary Distribution Desktop of Windows
 - locked CDROM / diskette distribution
 - any of 14 languages including Japanese
 - robust uninstall
 - on-line help
 - more



917-C Willowbrook Drive • Huntsville, AL 35802 • USA • 205.880.8782
 Call today! 800.448.4154 • BBS Demos 205.880.8785 • FAX info 205.880.8705
 From your FAX phone • MC/VISA/AMEX/PO • Internet sales@instalit.com

NEW!

INSTALIT/Crypto
hpiOMni/Crypto
INSTALIT V5.0
 DOS & OS/2 or Windows
INSTALIT/NT
INSTALIT/VBX
hpiWarehaus V2.0
hpiVend
Quick Release

Join us at our HPI User's Group Meeting
 "Emerging Distribution Technologies"
 Call for reservations! May 19-20 1994

¹ Modem and credit card required

² Try the demos, then purchase on the spot!

CIRCLE NO. 909 ON READER SERVICE CARD




```

for (i = N + 1; i > 1; --i) {
    Xl = Xl ^ P[i];
    Xr = F(Xl) ^ Xr;

    /* Exchange Xl and Xr */
    temp = Xl;
    Xl = Xr;
    Xr = temp;
}
/* Exchange Xl and Xr */
temp = Xl;
Xl = Xr;
Xr = temp;

Xr = Xr ^ P[1];
Xl = Xl ^ P[0];

*xl = Xl;
*xr = Xr;
}
short InitializeBlowfish(char key[], short keybytes)
{
    short i;
    short j;
    short k;
    short error;
    short numread;
    unsigned long data;
    unsigned long datal;
    unsigned long datar;
    /* First, open the file containing the array initialization data */
    error = opensubkeyfile();
    if (error == noErr) {
        for (i = 0; i < N + 1; ++i) {
            numread = fread(&data, 4, 1, SubkeyFile);
            printf("%d : %d : %.4s\n", numread, i, &data);
            if (numread != 1) {
                return DATAERROR;
            } else {
                P[i] = data;
            }
        }
        for (i = 0; i < 4; ++i) {
            for (j = 0; j < 256; ++j) {
                numread = fread(&data, 4, 1, SubkeyFile);
                printf("%d, %d : %.4s\n", i, j, &data);
                if (numread != 1) {
                    return DATAERROR;
                } else {
                    S[i, j] = data;
                }
            }
        }
    }
}

```

```

}
fclose(SubkeyFile);
j = 0;
for (i = 0; i < 18; ++i) {
    data = 0x00000000;
    for (k = 0; k < 4; ++k) {
        data = (data << 8) | key[j];
        j = j + 1;
        if (j > keybytes) {
            j = 0;
        }
    }
    P[i] = P[i] ^ data;
}
datal = 0x00000000;
datar = 0x00000000;
for (i = 0; i < 18; i += 2) {
    Blowfish_encrypter(&datal, &datar);

    P[i] = datal;
    P[i + 1] = datar;
}
for (j = 0; j < 4; ++j) {
    for (i = 0; i < 256; i += 2) {
        Blowfish_encrypter(&datal, &datar);

        S[j, i] = datal;
        S[j, i + 1] = datar;
    }
}
} else {
    printf("Unable to open subkey initialization file : %d\n", error);
}
return error;
}

```

End Listings

\$325 RISC Module

Eliminate PCs from Control and Display Applications.
32 Bit ARM RISC with 64 bit SVGA out performs 486!

The Pixel Press module from Applied Data Systems brings 32 bit ARM RISC processing power to Machine Control, Data Processing, Status Display and IO control applications. Unlike Intel processors, the ARM processor "Boots" in 32 bit mode and requires no memory management or operating systems overhead. Any programmer with Z80 or 8051 experience will appreciate this simple, 32 bit wide and fast (83 ns cycle) architecture.

While users can write custom C or Assembly programs, many applications will use the ROMed graphic commands for display creation and control. Direct connection to either a parallel port or processor bus allows high speed (1 MByte/Sec) transfers of commands and data. Downloaded user code can control over 64 ports of external user hardware.

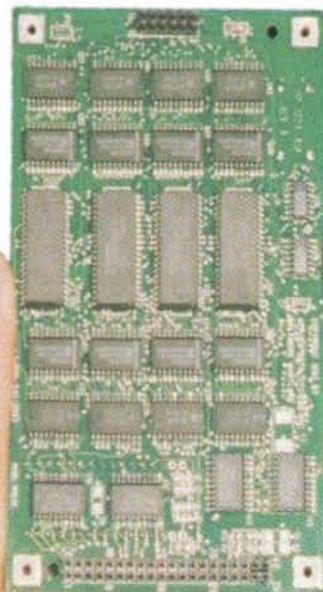
Each Pixel Press module contains 256K Bytes of processor DRAM, 512K Bytes of memory mapped (packed pixels) frame buffer DRAM, upto 4MB of EPROM, a serial debug port, and a voltage supervisor / watchdog timer circuit. A Xilinx FPGA provides either CGA, VGA or SVGA video (1024 x 768 None Interlaced) at 4 bits per pixel and supports LCD, EL and Plasma display technologies. The 3x5x0.5 inch size and 800ma @ 5V power requirement make the Pixel Press ideal for embedded applications.

A Demo Kit consisting of the Pixel Press, SVGA monitor, demo circuit and power supply is available for only \$795. A C Compiler/Assembler/Debugger is available FREE!

For More Information Call:

Applied Data Systems, Inc. 409A East Preston St. Baltimore MD 21202 USA
1-800-541-2003 FAX 1-410-576-0338 Outside USA 1-410-576-0335

CIRCLE NO. 915 ON READER SERVICE CARD



Listing One (Text begins on page 44.)

```

/* CONVOLVE.C */
/* Copyright (C) 1993 Mac A. Cody - All rights reserved */

#include "convolve.h"

/* DualConvDec2 - Convolution of data array with decomposition filters
   followed by decimation by two.
   Input(s): REAL_TYPE *src - Pointer to source data sample array.
             REAL_TYPE *htilda - Pointer to lowpass filter array.
             REAL_TYPE *gtilda - Pointer to highpass filter array.
             short srclen - length of source data array.
             short filten - length of filter arrays.
   Output(s): REAL_TYPE *adst - Pointer to approximation data sample array.
             REAL_TYPE *ddst - Pointer to detail data sample array.
*/
void DualConvDec2(REAL_TYPE *src, REAL_TYPE *adst, REAL_TYPE *ddst,
                  REAL_TYPE *htilda, REAL_TYPE *gtilda, short srclen, short filten)
{
    short i, j, brklen, endlen;
    REAL_TYPE adot_p, ddot_p;
    REAL_TYPE *head_src, *lp_fltr, *hp_fltr;
    brklen = 1; /* initial break in dot product is after first element */
    /* perform truncated dot products until filter no longer hangs off end of
       array: decimation by two handled by two-element shift: break count
       increases by two on every iteration */
    for(j = 0; j < filten; j += 2)
    {
        head_src = src + j; /* point to appropriate initial element at head */
        lp_fltr = htilda; /* set up pointer to lowpass filter */
        hp_fltr = gtilda; /* set up pointer to highpass filter */
        adot_p = *head_src * *lp_fltr++; /* initial lowpass product of head */
        ddot_p = *head_src-- * *hp_fltr++; /* initial highpass product of head */
        for(i = 1; i < brklen; i++) /* perform remaining products of head */
        {
            adot_p += *head_src * *lp_fltr++;
            ddot_p += *head_src-- * *hp_fltr++;
        }
        *adst++ = adot_p; /* save the completed lowpass dot product */
        *ddst++ = ddot_p; /* save the completed highpass dot product */
    }
    endlen = srclen + filten - 2; /* find total length of array */
    /* perform convolution to the physical end of the array
       with a simple dot product loop */
    for(j = 0; j <= endlen; j += 2)
    {
        head_src = src + j; /* point to appropriate initial element at head */
        lp_fltr = htilda; /* set up pointer to lowpass filter */
        hp_fltr = gtilda; /* set up pointer to highpass filter */
        adot_p = *head_src * *lp_fltr++; /* initial lowpass product */
        ddot_p = *head_src-- * *hp_fltr++; /* initial highpass product */
        for(i = 1; i < filten; i++) /* perform remaining products */
        {
            adot_p += *head_src * *lp_fltr++;
            ddot_p += *head_src-- * *hp_fltr++;
        }
        *adst++ = adot_p; /* save the completed lowpass dot product */
        *ddst++ = ddot_p; /* save the completed highpass dot product */
    }
    /* perform convolution off the physical end of the array
       with a partial dot product loop, like at the beginning */
    for(brklen = filten - 2, j = 2; brklen > 0; brklen -= 2, j += 2)
    {
        head_src = src + endlen; /* point to last element in array */
        lp_fltr = htilda + j; /* set up pointer to lowpass filter offset */
        hp_fltr = gtilda + j; /* set up pointer to highpass filter offset */
        adot_p = *head_src * *lp_fltr++; /* initial lowpass product */
        ddot_p = *head_src-- * *hp_fltr++; /* initial highpass product */
        for(i = 1; i < brklen; i++) /* perform remaining products */
        {
            adot_p += *head_src * *lp_fltr++;
            ddot_p += *head_src-- * *hp_fltr++;
        }
        *adst++ = adot_p; /* save the completed lowpass dot product */
        *ddst++ = ddot_p; /* save the completed highpass dot product */
    }
} /* End of DualConvDec2 */

/* DualConvInt2Sum - Convolution of data array with reconstruction
   filters with interpolation by two and sum together.
   Input(s): REAL_TYPE *asrc - Pointer to approximation data sample array.
             REAL_TYPE *dsrc - Pointer to detail data sample array.
             REAL_TYPE *h - Pointer to lowpass filter array.
             REAL_TYPE *g - Pointer to highpass filter array.
             short srclen - length of source data array.
             short filten - length of filter arrays.
   Output(s): REAL_TYPE *dst - Pointer to output data sample array.
*/
void DualConvInt2Sum(REAL_TYPE *asrc, REAL_TYPE *dsrc, REAL_TYPE *dst,
                    REAL_TYPE *h, REAL_TYPE *g, short srclen, short filten)
{
    short i, j, endlen;
    REAL_TYPE dot_pe, dot_po;
    REAL_TYPE *head_asrc, *head_dsrc, *lp_fltr, *hp_fltr;
    endlen = srclen + filten - 2; /* find total length of array */
    filten /= 2; /* adjust filter length value for interpolation */
    j = filten - 1; /* start with filter 'covering' end of array */
    head_asrc = asrc + j; /* point to initial element at head */
    head_dsrc = dsrc + j; /* point to initial element at head */
    lp_fltr = h + 1; /* set up pointer to lowpass filter */
    hp_fltr = g + 1; /* set up pointer to highpass filter */
    /* initial lowpass and highpass odd product */
    dot_po = *head_asrc-- * *lp_fltr + *head_dsrc-- * *hp_fltr;
    lp_fltr += 2; hp_fltr += 2; /* skip over even filter elements */
    for(i = 1; i < filten; i += 2) /* perform remaining products */
    {
        dot_po += *head_asrc-- * *lp_fltr + *head_dsrc-- * *hp_fltr;
        lp_fltr += 2; hp_fltr += 2; /* skip over even filter elements */
    }
    /* save the completed lowpass and highpass odd dot product */
    *dst++ = dot_po;
    /* perform initial convolution with a simple dot product loop */

```

```

for(j++; j <= endlen; j++)
{
    head_asrc = asrc + j; /* point to appropriate initial element at head */
    head_dsrc = dsrc + j; /* point to appropriate initial element at head */
    lp_fltr = h; /* set up pointer to lowpass filter */
    hp_fltr = g; /* set up pointer to highpass filter */
    /* initial lowpass and highpass even product */
    dot_pe = *head_asrc * *lp_fltr++ + *head_dsrc * *hp_fltr++;
    /* initial lowpass and highpass odd product */
    dot_po = *head_asrc-- * *lp_fltr++ + *head_dsrc-- * *hp_fltr++;
    for(i = 1; i < filten; i++) /* perform remaining products */
    {
        dot_pe += *head_asrc * *lp_fltr++ + *head_dsrc * *hp_fltr++;
        dot_po += *head_asrc-- * *lp_fltr++ + *head_dsrc-- * *hp_fltr++;
    }
    /* save the completed lowpass and highpass even dot product */
    *dst++ = dot_pe;
    /* save the completed lowpass and highpass odd dot product */
    *dst++ = dot_po;
}
} /* End of DualConvInt2Sum */

```

Listing Two

```

/* AWPT.C */
/* Copyright (C) 1993 Mac A. Cody - All rights reserved */

#include "wp_types.h"
#include "awpt.h"
#include "convolve.h"

/* AWPT - Aperiodic Wavelet Packet Transform: Data is assumed to be
   non-periodic, so convolutions do not wrap around arrays.
   Convolution data past end of data is generated and retained
   to allow perfect reconstruction of original input.
   Input(s): REAL_TYPE *indata - Pointer to input data sample array.
             REAL_TYPE *htilda - Pointer to lowpass filter array.
             REAL_TYPE *gtilda - Pointer to highpass filter array.
             short filten - Length of filter arrays.
   Output(s): WPTstruct *out - Pointer to transform data structure.
   Note: Structure pointed to by 'out' contains:
         out->levels - Number of levels in transform (short).
         out->length - Length of input data sample array (short).
         out->data - Pointer to pointer of arrays of data (REAL_TYPE ***).
*/
void AWPT(REAL_TYPE *indata, WPTstruct *out,
          REAL_TYPE *htilda, REAL_TYPE *gtilda, short filten)
{
    short i, j, nodes, datalen;
    REAL_TYPE *src;
    levels = out->levels;
    datalen = out->length; /* start with length of input array */
    /* loop for all levels, halving the data length on each lower level */
    for(i = 0, nodes = 2; i < levels; i++, nodes *= 2)
    {
        for(j = 0; j < nodes; j += 2)
        {
            if(out->data[i][j] == 0) continue;
            if(i == 0) /* ... source for highest level is input data */
                src = indata;
            else /* ... source is corresponding node of higher level */
                src = out->data[i-1][j >> 1];
            DualConvDec2(src, out->data[i][j], out->data[i][j+1],
                        htilda, gtilda, datalen, filten);
        }
        datalen /= 2; /* input length for next level is half this level */
    }
} /* End of AWPT */

/* IAWPT - Inverse Aperiodic Wavelet Packet Transform: Data is assumed to be
   non-periodic, so convolutions do not wrap around arrays.
   Convolution data past end of data is used to generate perfect
   reconstruction of original input.
   Input(s): WPTstruct *in - Pointer to transform data structure.
             REAL_TYPE *htilda - Pointer to lowpass filter array.
             REAL_TYPE *gtilda - Pointer to highpass filter array.
             short filten - Length of filter arrays.
   Note: Structure pointed to by 'in' contains:
         in->levels - Number of levels in transform (short).
         in->length - Length of output data sample array (short).
         in->data - Pointer to pointer of arrays of data (REAL_TYPE ***).
   Output(s): REAL_TYPE *indata - Pointer to input data sample array.
*/
void IAWPT(WPTstruct *in, REAL_TYPE *outdata,
           REAL_TYPE *htilda, REAL_TYPE *gtilda, short filten)
{
    short i, j, levels, nodes, datalen;
    REAL_TYPE *dst;
    levels = in->levels;
    /* start with length of lowest level input array */
    datalen = in->length >> levels;
    /* loop for all levels, doubling the data length on each higher level:
       destination of all but the highest branch of the reconstruction
       is the next higher node */
    for(i = levels - 1, nodes = 1 << levels; i >= 0; i--, nodes *= 2)
    {
        for(j = 0; j < nodes; j += 2)
        {
            if(out->data[i][j] == 0) continue;
            if(i == 0) /* ... destination for highest level is input data */
                dst = outdata;
            else /* ... destination is corresponding node of higher level */
                dst = in->data[i-1][j >> 1];
            DualConvInt2Sum(in->data[i][j], in->data[i][j+1], dst,
                          htilda, gtilda, datalen, filten);
        }
        datalen *= 2; /* input length for next level is half this level */
    }
} /* End of IAWPT */

```

End Listings

Listing One (Text begins on page 56.)

```

/* Update to Greg Viot's fuzzy system -- DDJ, February 1993, page 94 */
/* By J. Tucker, P. Fraley, and L. Swanson, April 1993 */
#include <stdio.h>
#include <string.h>
#define max(a,b) (a<b ? b : a)
#define min(a,b) (a>b ? b : a)
struct io_type *System_Inputs;
struct io_type *System_Output;
#define MAXNAME 10
#define UPPER_LIMIT 255
struct io_type {
    char name[MAXNAME];
    int value;
    struct mf_type *membership_functions;
    struct io_type *next;
};
struct mf_type {
    char name[MAXNAME];
    int value;
    int point1;
    int point2;
    float slope1;
    float slope2;
    struct mf_type *next;
};
struct rule_type {
    struct rule_element_type *if_side;
    struct rule_element_type *then_side;
    struct rule_type *next;
};
struct rule_element_type {
    int *value;
    struct rule_element_type *next;
};
struct rule_type *Rule_Base;

main(argc,argv)
int argc;
char *argv[];
{
    int input1, input2;
    if(argc==3)
    {
        printf("Error - Must supply 2 numeric inputs.\n");
        printf("Inputs scaled to range 0-255.\n");
        printf("Usage: %s angle velocity\n",argv[0]);
        exit(0);
    }
    input1=atoi(argv[1]);
    input2=atoi(argv[2]);
    initialize_system();
    get_system_inputs(input1,input2);
    fuzzification();
    rule_evaluation();
    defuzzification();
    put_system_outputs();
}

fuzzification()
{
    struct io_type *si;
    struct mf_type *mf;
    for(si=System_Inputs;si!=NULL;si=si->next)
        for(mf=si->membership_functions;mf!=NULL;mf=mf->next)
            compute_degree_of_membership(mf,si->value);
}

rule_evaluation()
{
    struct rule_type *rule;
    struct rule_element_type *ip;
    struct rule_element_type *tp;
    int strength;
    int nomatch=0;
    for(rule=Rule_Base;rule!=NULL;rule=rule->next)
    {
        strength=UPPER_LIMIT;
        for(ip=rule->if_side;ip!=NULL;ip=ip->next)
            strength=min(strength,*ip->value);
        for(tp=rule->then_side;tp!=NULL;tp=tp->next)
            strength=min(strength,*tp->value);
        if(strength>0)nomatch=1;
    }
    if(nomatch==0)printf("NO MATCHING RULES FOUND!\n");
}

defuzzification()
{
    struct io_type *so;
    struct mf_type *mf;
    int sum_of_products;
    int sum_of_areas;
    int area, centroid;
    for(so=System_Output;so!=NULL;so=so->next)
    {
        sum_of_products=0;
        sum_of_areas=0;
        for(mf=so->membership_functions;mf!=NULL;mf=mf->next)
        {
            area=compute_area_of_trapezoid(mf);
            centroid=mf->point1*(mf->point2-mf->point1)/2;
            sum_of_products+=area*centroid;
            sum_of_areas+=area;
        }
        if(sum_of_areas==0)
        {
            printf("Sum of Areas = 0, will cause div error\n");
            printf("Sum of Products= %d\n",sum_of_products);
            so->value=0;
            return;
        }
        so->value=sum_of_products/sum_of_areas;
    }
}

compute_degree_of_membership(mf,input)
struct mf_type *mf;
int input;
{
    int delta_1, delta_2;
    delta_1=input - mf->point1;
    delta_2=mf->point2 - input;
    if((delta_1<=0)||((delta_2>0)&&mf->value==0))

```

```

else
{
    mf->value=min((mf->slope1*delta_1),(mf->slope2*delta_2));
    mf->value=min(mf->value,UPPER_LIMIT);
}
}

compute_area_of_trapezoid(mf)
struct mf_type *mf;
{
    float run_1,run_2,area,top;
    float base;
    base=mf->point2 - mf->point1;
    run_1=mf->value / mf->slope1;
    run_2=mf->value / mf->slope2;
    top=base - run_1 - run_2;
    area=mf->value*(base+top)/2;
    return(area);
}

initialize_system()
{
    int a, b, c, d, x;
    char buff[10],buff1[4],buff2[4];
    static char filename1[]="in1";
    static char filename2[]="in2";
    static char filename3[]="out1";
    FILE *fp;
    struct io_type *outptr;
    struct mf_type *top_mf;
    struct mf_type *mfptr;
    struct io_type *ioptr;
    struct rule_type *ruleptr;
    struct rule_element_type *ifptr;
    struct rule_element_type *thenptr;
    ioptr=NULL;
    ruleptr=NULL;
    ifptr=NULL;
    thenptr=NULL;
}

/* READ THE FIRST FUZZY SET (ANTECEDENT); INITIALIZE STRUCTURES */
if((fp=fopen(filename1,"r"))==NULL)
{
    printf("ERROR- Unable to open data file named %s.\n",filename1);
    exit(0);
}
ioptr=(struct io_type *)calloc(1,sizeof(struct io_type));
System_Inputs=ioptr;
x=fscanf(fp,"%e",buff);
sprintf(ioptr->name,"%e",buff);
mfptr=NULL;
while((x=fscanf(fp,"%e %d %d %d",buff,&a,&b,&c,&d))!=EOF)
{
    if(mfptr==NULL)
    {
        mfptr=(struct mf_type *)calloc(1,sizeof(struct mf_type));
        top_mf=mfptr;
        ioptr->membership_functions=mfptr;
    }
    else
    {
        for(mfptr=top_mf;mfptr->next;mfptr=mfptr->next);
    }
}

```

(continued on page 102)

TLIB™ Version Control

For DOS, OS/2 and Windows-NT

• The experts loved TLIB 4:

"...amazingly fast... TLIB is a great system." **PC Tech Journal**
 "TLIB has features and power to spare... TLIB is easy to use and the fastest of the reviewed packages." **Computer Language**
 "I will not program without it." **Uptime Magazine**

• TLIB 5.01 adds:

Automatic branching. Automatic version labeling across branches. User defined **promote** structures, for staged development. Exclusive whole-level **change migration** for customized software. N-way-tree version numbers. Branch and full locking. OS/2 & NT support. And now... **automated conversion from PVCS™ or MKS RCS!**

• Plus the features they loved in TLIB 4:

Check-in/out locking. Branching, for parallel development. Keywords. Full binary file support (does not depend upon NLS in the file like other products). Wildcard and list-of-file support; can create lists by scanning source code for includes. Can merge (reconcile) multiple simultaneous changes and undo intermediate revisions. Network and WORM optical disk support. Mainframe-compatible delta generator for Pansophic, ADR, IBM, Sperry formats. Integrated with industry-leading MAKE from Opus™ software.

MS-DOS \$139, OS/2 & NT (with MS-DOS) \$195 + shipping.
 5-user net: DOS \$419, OS/2+NT+DOS \$595. Call for other sizes.



Burton Systems Software

PO Box 4156, Cary, NC 27519 (919) 233-8128

FAX: 233-0716

See us at Software Development '94 in booth #1918

CIRCLE NO. 247 ON READER SERVICE CARD

Listing One (Listing continued, text begins on page 56.)

```

    mfptr->next=(struct mf_type *)calloc(1,sizeof(struct mf_type));
    mfptr=mfptr->next;
}
sprintf(mfptr->name,"%s",buff); /* membership name, NL, ZE, etc */
mfptr->point1=a; /* left x axis value */
mfptr->point2=d; /* right x axis value */
if(b-a>0) mfptr->slope1=UPPER_LIMIT/(b-a); /* left slope */
else
{
    printf("Error in input file %s, membership element %s.\n",
        filename1,buff);
    exit(1);
}
if(d-c>0) mfptr->slope2=UPPER_LIMIT/(d-c); /* right slope */
else
{
    printf("Error in input file %s, membership element %s.\n",
        filename1,buff);
    exit(1);
}
}
close(fp); /* close "angles" file */
/* READ THE SECOND FUZZY SET (ANTECEDENT): INITIALIZE STRUCTURES */
if((fp=fopen(filename2,"r"))==NULL) /* open "velocity" file */
{
    printf("ERROR- Unable to open data file named %s.\n",filename2);
    exit(0);
}
ioptr->next=(struct io_type *)calloc(1,sizeof(struct io_type));
ioptr=ioptr->next;
x=fscanf(fp,"%s",buff); /* from 1st line, get set's name */
sprintf(ioptr->name,"%s",buff); /* into struct io_type.name */
mfptr=NULL;
while((x=fscanf(fp,"%s %d %d %d",buff,&a,&b,&c,&d))!=EOF) /* get line */
{
    if(mfptr==NULL) /* first time thru only */
    {
        mfptr=(struct mf_type *)calloc(1,sizeof(struct mf_type));
        top_mf=mfptr;
        ioptr->membership_functions=mfptr;
    }
    else
    {
        for(mfptr=top_mf;mfptr->next;mfptr=mfptr->next); /* spin to last */
        mfptr->next=(struct mf_type *)calloc(1,sizeof(struct mf_type));
        mfptr=mfptr->next;
    }
    sprintf(mfptr->name,"%s",buff); /* membership name, NL, ZE, etc */
    mfptr->point1=a; /* left x axis value */
    mfptr->point2=d; /* right x axis value */
    if(b-a>0) mfptr->slope1=UPPER_LIMIT/(b-a); /* left slope */
    else
    {
        printf("Error in input file %s, membership element %s.\n",
            filename2,buff);
        exit(1);
    }
    if(d-c>0) mfptr->slope2=UPPER_LIMIT/(d-c); /* right slope */
    else
    {
        printf("Error in input file %s, membership element %s.\n",
            filename2,buff);
        exit(1);
    }
}
close(fp); /* close "velocity" file */
/* READ THE THIRD FUZZY SET (CONSEQUENCE): INITIALIZE STRUCTURES */
if((fp=fopen(filename3,"r"))==NULL) /* open "force" file */
{
    printf("ERROR- Unable to open data file named %s.\n",filename3);
    exit(0);
}
ioptr=(struct io_type *)calloc(1,sizeof(struct io_type));
System_Output=ioptr;
/* Anchor output structure */
x=fscanf(fp,"%s",buff); /* from 1st line, get set's name */
sprintf(ioptr->name,"%s",buff); /* into struct io_type.name */
mfptr=NULL;
while((x=fscanf(fp,"%s %d %d %d",buff,&a,&b,&c,&d))!=EOF) /* get line */
{
    if(mfptr==NULL) /* first time thru */
    {
        mfptr=(struct mf_type *)calloc(1,sizeof(struct mf_type));
        top_mf=mfptr;
        ioptr->membership_functions=mfptr;
    }
    else
    {
        for(mfptr=top_mf;mfptr->next;mfptr=mfptr->next);
        mfptr->next=(struct mf_type *)calloc(1,sizeof(struct mf_type));
        mfptr=mfptr->next;
    }
    sprintf(mfptr->name,"%s",buff); /* membership name, NL, ZE, etc */
    mfptr->point1=a; /* left x axis value */
    mfptr->point2=d; /* right x axis value */
    if(b-a>0) mfptr->slope1=UPPER_LIMIT/(b-a); /* left slope */
    else
    {
        printf("Error in input file %s, membership element %s.\n",
            filename3,buff);
        exit(1);
    }
    if(d-c>0) mfptr->slope2=UPPER_LIMIT/(d-c); /* right slope */
    else
    {
        printf("Error in input file %s, membership element %s.\n",
            filename3,buff);
        exit(1);
    }
}
close(fp); /* close "force" file */
/* READ RULES FILE: INITIALIZE STRUCTURES */
ioptr=NULL;
outptr=NULL;
if((fp=fopen("rules","r"))==NULL) /* open rules file */
{
    printf("ERROR- Unable to open data file named %s.\n","rules");
    exit(0);
}
ruleptr=(struct rule_type *)calloc(1,sizeof(struct rule_type));
if(ioptr==NULL) Rule_Base=ruleptr; /* first time thru, anchor */
while((x=fscanf(fp,"%s %s %s",buff,buff1,buff2))!=EOF) /* get a line */
{
    ioptr=System_Inputs; /* points to angle */
    for(mfptr=iopt->membership_functions;mfptr!=NULL;mfptr=mfptr->next)

```

```

    {
        if((strcmp(mfptr->name,buff))!=0)
        {
            ifptr=(struct rule_element_type *)
                calloc(1,sizeof(struct rule_element_type));
            ruleptr->if_side=ifptr; /* points to angle */
            ifptr->value=&mfptr->value; /* needs address here */
            ifptr->next=(struct rule_element_type *)
                calloc(1,sizeof(struct rule_element_type));
            ifptr=ifptr->next; /* match found */
            break;
        }
    }
    ioptr=iopt->next; /* points to velocity */
    for(mfptr=iopt->membership_functions;mfptr!=NULL;mfptr=mfptr->next)
    {
        if((strcmp(mfptr->name,buff1))!=0)
        {
            ifptr->value=&mfptr->value; /* needs address here */
            break; /* match found */
        }
    }
    if(outptr==NULL) outptr=System_Output; /* point then stuff to output */
    for(mfptr=outptr->membership_functions;mfptr!=NULL;mfptr=mfptr->next)
    {
        if((strcmp(mfptr->name,buff2))!=0)
        {
            thenptr=(struct rule_element_type *)
                calloc(1,sizeof(struct rule_element_type));
            ruleptr->then_side=thenptr;
            thenptr->value=&mfptr->value; /* needs address here */
            break; /* match found */
        }
    }
    ruleptr->next=(struct rule_type *)calloc(1,sizeof(struct rule_type));
    ruleptr=ruleptr->next;
}
close(fp); /* END WHILE READING RULES FILE */
/* close "rules" file */
/* END INITIALIZE */
put_system_outputs()
{
    struct io_type *iopt;
    struct mf_type *mfp;
    struct rule_type *ruleptr;
    struct rule_element_type *ifptr;
    struct rule_element_type *thenptr;
    int cnt=1;
    for(iopt=System_Inputs;iopt!=NULL;iopt=iopt->next)
    {
        printf("%s: Value= %d\n",iopt->name,iopt->value);
        for(mfp=iopt->membership_functions;mfp!=NULL;mfp=mfp->next)
        {
            printf(" %s: Value %d Left %d Right %d\n",
                mfp->name,mfp->value,mfp->point1,mfp->point2);
        }
        printf("\n");
    }
    for(iopt=System_Output;iopt!=NULL;iopt=iopt->next)
    {
        printf("%s: Value= %d\n",iopt->name,iopt->value);
        for(mfp=iopt->membership_functions;mfp!=NULL;mfp=mfp->next)
        {
            printf(" %s: Value %d Left %d Right %d\n",
                mfp->name,mfp->value,mfp->point1,mfp->point2);
        }
    }
    /* print values pointed to by rule_type (if & then) */
    printf("\n");
    for(ruleptr=Rule_Base;ruleptr->next!=NULL;ruleptr=ruleptr->next)
    {
        printf("Rule %d: cnt++");
        for(ifptr=ruleptr->if_side;ifptr!=NULL;ifptr=ifptr->next)
            printf(" %d",*(ifptr->value));
        for(thenptr=ruleptr->then_side;thenptr!=NULL;thenptr=thenptr->next)
            printf(" %d\n",*(thenptr->value));
    }
    printf("\n");
}
/* END PUT SYSTEM OUTPUTS */
get_system_inputs(input1,input2)
/* NEW */
{
    struct io_type *iopt;
    iopt=System_Inputs;
    iopt->value=input1;
    iopt->next;
    iopt->value=input2;
}
/* END GET SYSTEM INPUTS */

```

End Listing

Listing One (Text begins on page 64.)

```
//-----
// PPDI0 Parallel Port Digital IO routines
// Version 1.0 Copyright 1993 by Brian Hook. All Rights Reserved.
// File: PPDI0.H -- header file for the PPDI0 library
// Compile with Borland C++ 3.1 -- porting to another compiler
// should be extremely trivial.
//-----
#ifndef __PPDI0_H
#define __PPDI0_H

//--- Pin definitions for control register ---
#define PIN_1      0x01
#define PIN_14     0x02
#define PIN_16     0x04
#define PIN_17     0x08

//--- Pin definitions for status register ---
#define PIN_15     0x08
#define PIN_13     0x10
#define PIN_12     0x20
#define PIN_10     0x40
#define PIN_11     0x80

//--- Interrupt enable bit definition ---
#define PTR_ENABLE_INT_BIT 0x10

//--- Function prototypes ---
unsigned PPDI0_GetLptAddress( int lpt_port );
void PPDI0_InstallISR( void interrupt (*fnc)(), int irq );
unsigned char PPDI0_ReadControlRaw( void );
unsigned char PPDI0_ReadStatusRaw( void );
unsigned char PPDI0_ReadControlCooked( void );
unsigned char PPDI0_ReadStatusCooked( void );
void PPDI0_RemoveISR( void );
void PPDI0_SendByte( unsigned char data );
void PPDI0_SetBaseAddress( unsigned base_address );
void PPDI0_SetLptPort( int lpt_port );

#endif
```

Listing Two

```
//-----
// PPDI0 Parallel Port Digital IO routines
// Version 1.0 Copyright 1993 by Brian Hook. All Rights Reserved.
// File: PPDI0.C -- code and variables for the PPDI0 library
// Compile with Borland C++ 3.1 -- porting to another compiler
// should be extremely trivial.
//-----
#include <dos.h>
#include "ppdio.h"

static unsigned ppdio_data_register;
static unsigned ppdio_control_register;
static unsigned ppdio_status_register;
static unsigned ppdio_interrupt_no;
static unsigned ppdio_irq;

static unsigned char ppdio_old_control_value;
static unsigned char ppdio_old_8259_mask;
static void interrupt (*ppdio_old_intvec)();

unsigned PPDI0_GetLptAddress( int lpt_no )
{
    unsigned far *pp = ( unsigned far * ) MK_FP( 0x40, 8 );
    //--- Assumes values of 1, 2, or 3 ---
    return ( pplpt_no-1 );
}

void PPDI0_InstallISR( void interrupt (*fnc)(), int irq_no )
{
    static char mask[] = { 0xf0, 0xf7, 0xef, 0xdf, 0xaf, 0x7f };
    unsigned char temp;

    //--- Interrupt number = IRQ no + 8 ---
    ppdio_interrupt_no = irq_no + 8;

    //--- Save original interrupt vector ---
    ppdio_old_intvec = getvect( ppdio_interrupt_no );

    //--- Install new ISR ---
    setvect( ppdio_interrupt_no, fnc );

    //--- Enable interrupts by setting the PTR_ENABLE_INT_BIT in
    //--- the control register. Also, OR it by 0x04 to send pin
    //--- 16 high then write out a 0 to pins 1, 14, and 17 so
    //--- that we can use the control register for input.
    ppdio_old_control_value = inportb( ppdio_control_register );
    temp = ppdio_old_control_value | PTR_ENABLE_INT_BIT | PIN_16;
    temp &= ~ ( PIN_17 | PIN_14 | PIN_1 );
    outportb( ppdio_control_register, temp );

    //--- Unmask our IRQ in the interrupt controller ---
    ppdio_old_8259_mask = inportb( 0x21 );
    temp = ppdio_old_8259_mask & mask[ppdio_interrupt_no-10];
    outportb( 0x21, temp );

    //--- Clear pending interrupts ---
    outportb( 0x20, 0x20 );
}

unsigned char PPDI0_ReadControlCooked( void )
{
    unsigned char raw_control;
    unsigned char cooked_control = 0;
    raw_control = PPDI0_ReadControlRaw();

    //--- Return a control register mask that compensates for the inverse logic
    //--- of pins 1, 14, and 17, and with 0s where bits are reserved or unused.
    if ( !( raw_control & PIN_1 ) )
```

```
        cooked_control |= PIN_1;
    if ( !( raw_control & PIN_14 ) )
        cooked_control |= PIN_14;
    if ( raw_control & PIN_16 )
        cooked_control |= PIN_16;
    if ( !( raw_control & PIN_17 ) )
        cooked_control |= PIN_17;
    return ( cooked_control );
}

unsigned char PPDI0_ReadControlRaw( void )
{
    return ( inportb( ppdio_control_register ) );
}

unsigned char PPDI0_ReadStatusCooked( void )
{
    unsigned char raw_status;
    unsigned char cooked_status = 0;

    raw_status = PPDI0_ReadStatusRaw();
    //--- Return a status register mask that compensates for the
    //--- inverse logic of pin 11, and with 0s for any reserved or unused bits.
    if ( raw_status & PIN_15 )
        cooked_status |= PIN_15;
    if ( raw_status & PIN_13 )
        cooked_status |= PIN_13;
    if ( raw_status & PIN_12 )
        cooked_status |= PIN_12;
    if ( !( raw_status & PIN_11 ) )
        cooked_status |= PIN_11;
    return ( cooked_status );
}

unsigned char PPDI0_ReadStatusRaw( void )
{
    return ( inportb( ppdio_status_register ) );
}

void PPDI0_RemoveISR( void )
{
    //--- Restore the interrupt controller's previous state ---
    outportb( 0x21, ppdio_old_8259_mask );

    //--- Restore the original interrupt vector ---
    setvect( ppdio_interrupt_no, ppdio_old_intvec );

    //--- Restore the printer control register ---
    outportb( ppdio_control_register, ppdio_old_control_value );
}

void PPDI0_SendByte( unsigned char data )
{
    outportb( ppdio_data_register, data );
}

void PPDI0_SetBaseAddress( unsigned base_address )
{
    ppdio_data_register = base_address;
    ppdio_status_register = base_address + 1;
    ppdio_control_register = base_address + 2;
}

void PPDI0_SetLptPort( int lpt_port )
{
    PPDI0_SetBaseAddress( PPDI0_GetLptAddress( lpt_port ) );
}
```

Listing Three

```
//-----
// PPDI0 Parallel Port Digital IO routines
// Version 1.0 Copyright 1993 by Brian Hook. All Rights Reserved.
// File: DIO.C -- this is an example how you could use the PPDI0
// routines. This could be used as a framework upon which you
// could build real applications.
// Compile with Borland C++ 3.1 -- porting to another compiler
// should be extremely trivial.
//-----
#include <conio.h>
#include "ppdio.h"

volatile int isr_called = 0;
void huge interrupt MyISR( void )
{
    isr_called = 1;

    //--- Normally you would read the input pins here and do something important
    //--- Signal end of interrupt to the interrupt controller ---
    outportb( 0x20, 0x20 );
}

void main( void )
{
    //--- Use LPT1 ---
    PPDI0_SetLptPort( 1 );

    //--- Install our ISR on IRQ 5 ---
    PPDI0_InstallISR( MyISR, 5 );

    //--- Run until either a key is pressed or interrupt is generated on IRQ 5
    while ( !kbhit() && !isr_called ) {
        PPDI0_RemoveISR();
    }
}
```

End Listings

APRIL 19-21, 1994 ♦ HYNES CONVENTION CENTER ♦ BOSTON, MA

THE EMBEDDED SYSTEMS CONFERENCE IS COMING TO **BOSTON**

DESTINATION: BOSTON

The world's largest conference and exhibition dedicated to microprocessor- and microcontroller-based development returns East this spring for a command performance. Even if you attended our East Coast debut in Atlanta last year, Boston is where you should be April 19-21, 1994 — for the second annual **EMBEDDED SYSTEMS CONFERENCE EAST**.

WE SPEAK ALL THE LANGUAGES

The **EMBEDDED SYSTEMS CONFERENCE EAST** will help you hone the technical and project management skills required for the entire embedded process. If you want to improve design, write more efficient code, devise more effective project management strategies, and see the future of embedded development more clearly, we've got what you need.

TIPS, TRICKS, and METHODOLOGIES

Mix and match dozens of hands-on programming and methodology workshops, lectures, and tutorials on topics like these:

♦ Design methodology ♦ Debugging ♦ Languages ♦ Networks
♦ Chip programming ♦ Hardware interfacing ♦ Digital signal processing ♦ Project management ♦ Fuzzy logic

MAIL OR FAX TO:

EMBEDDED SYSTEMS CONFERENCE EAST *

Miller Freeman, Inc.
Attn: Julie Ann Lee
600 Harrison Street
San Francisco, CA 94107 USA

FAX (415) 905-2220 OR CALL (415) 905-2354

EXPERTS' INSIGHTS FIRSTHAND

Get direct access to dozens of today's top experts in embedded development, including:

- ♦ Paul Ward and Stephen Mellor, creators of the Ward-Mellor methodologies that brought CASE to real-time embedded systems
- ♦ Larry Constantine, co-author of the definitive text, *Structured Design*
- ♦ Plus many more



TARGETED PRODUCTS AND SERVICES

See the largest dedicated exhibition of embedded development tools and utilities, talk to representatives from more than 100 leading companies, and try the latest in:

- ♦ compilers/cross compilers ♦ in-circuit emulators ♦ logic analyzers
- ♦ single board computers ♦ microprocessors/ microcontrollers ♦ debugging tools ♦ real-time operating systems ♦ cross assemblers ♦ simulators ♦ and more . . .

SEND ME THE DETAILS FAST ON THE EMBEDDED SYSTEMS CONFERENCE EAST

April 19-21, 1994, Hynes Convention Center, Boston, MA

I'm interested in:

☐ Attending ☐ Exhibiting

Name _____

Title _____

Company _____

Address _____

City _____

State _____

Zip _____

Phone _____

Fax _____

ED14

Do you want more support in your day-to-day use of OS/2? Is your company struggling to integrate OS/2 into its enterprise computing strategy? Do you want to take OS/2 to a higher level?

OS/2 WORLD WILL POINT YOU IN THE RIGHT DIRECTION

The independent technical program, featuring over 100 lectures, workshops and tutorials, will present you with the objective truth — both good and bad — about OS/2, IBM, and third-party providers. You can count on the information because our top-notch faculty won't have a hidden agenda. More importantly, the technical program is geared to you, the OS/2 customer, not the industry.

The technical program is broken out into 8 dynamic tracks:

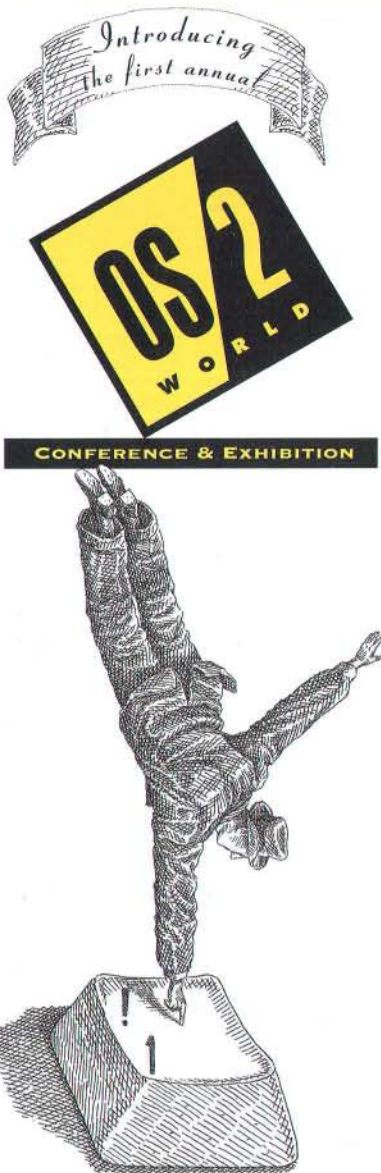
- Enterprise-Wide Networking
- Client/Server and Information Management
- Local Area Networking
- Power Computing
- Systems Administration
- Corporate Software Development
- Personal Programming
- Multimedia

AN OBJECTIVE AND BALANCED FACULTY

Classes will be taught almost exclusively by independent expert users, consultants, and trainers offering objective and balanced information to users and managers charged with deploying and maintaining OS/2 systems in corporate environments. Instructors will be experts who work with OS/2 in the field, and have implemented OS/2 solutions for their companies or clients.

INDUSTRY-WIDE SPONSORSHIP GUARANTEES A TOP QUALITY EVENT

Produced by Miller Freeman, Inc., the third largest trade show producer in North America, the OS/2 World Conference & Exhibition is presented in cooperation with the IBM Corporation and sponsored by OS/2 Magazine, OS/2 Developer, DBMS, Database Programming and Design, LAN Magazine, STACKS: The Network Journal, Software Development and Dr. Dobb's Journal.



July 19-22, 1994

**Santa Clara
Convention Center
Santa Clara, California**

CHOOSING THE RIGHT NEW PRODUCTS CAN BE QUITE A BALANCING ACT

The OS/2 World Products Exhibition is the only place you can get a hands-on look at the hardware and software that can make your system scream. You can pose your toughest questions as you meet face-to-face with the major players in the industry. This world-class exhibition will feature leading suppliers showcasing applications software, communications hardware and software, databases, network solutions, utilities, multimedia tools, and more. The exhibition will run Tuesday, Wednesday and Thursday so you will have plenty of time to get the answers you need.

SEE THE FUTURE OF OS/2 TODAY

Running parallel to the technical program will be dozens of presentations by key visionaries at IBM and major hardware and software providers. Topics include: OS/2 and Workplace OS, What It All Means To You • Customizing the Workplace Shell with REXX • Exploiting OS/2 Software Motion Video • Introduction To The OSE Distributed Computing Environment • And More.

Don't miss this opportunity to learn what applications and opportunities await OS/2 users of the future, and how you can stay on the cutting edge.

Phone, FAX or mail today for more information on the OS/2 World Conference & Exhibition.



YES! Please send me info on: ☐ Attending ☐ Exhibiting

NAME _____

TITLE _____

PHONE _____

COMPANY _____

FAX _____

ADDRESS _____

CITY _____

STATE/PROV. _____

ZIP CODE _____

OS/2 World Conference & Exhibition Attn: JulieAnn Lee
600 Harrison Street, 4th Floor, San Francisco, CA 94107-9603
Phone: (415) 905-2354 Fax: (415) 905-2220

DDOS1

Listing One (Text begins on page 92.)

```
//File: GRPHPHEN.H
#ifndef __GRPHPHEN_H
#define __GRPHPHEN_H

//Header for EOS class representing a phenotype.
//You need EOS v1.1 to compile this code
#ifdef __PHENO_H
#include "pheno.h"
#endif //__PHENO_H

class CGraphDrawingPheno : public TPhenotype
{
public:
    CGraphDrawingPheno(CGAPDriver &driver, int width, int height) ;
    ~CGraphDrawingPheno() ;
    double CalcFitness() ;
    void Decode(PTGenotype geno) ;
    PTPhenotype Copy() ;
    void GetPhenoInfo(void *pInfoStruct) ;
    void GetNearestEmptyCell(const int row, const int col, int &actualRow,
                             int &actualCol) ;

    BOOL Adjacent(WORD node1, WORD node2) ;
    BOOL Diagonal(WORD node1, WORD node2) ;
    BOOL FindNode(const WORD node, int &row, int &col) ;
    double Distance(WORD node1, WORD node2) ;
    double RectDistance(WORD node1, WORD node2) ;
private:
    int m_Width ;
    int m_Height ;
    CWordMatrix *m_pGrid ; //grid where each entry is a node
                             // number or EMPTY_CELL
    CGAPDriver &m_Driver ; //interface to the graph driver class
    int *m_GridIndex[2] ; //index into grid to quickly locate nodes
};
```

Listing Two

```
//File: GRPHPHEN.CPP
#include "stdafx.h"
//eos headers
#include "eos.h"
#include "eosutil.h"
#include "geno.h"

//graph GA headers
#include "grphphen.h"
#include "wmatrix.h"
#include "gdriver.h"
#include "grphutil.h"

const HIGHEST_REWARD = 10 ;
const MEDIUM_REWARD = 5 ;
const SMALLEST_REWARD = 1 ;
const HIGHEST_PENALTY = 10 ;
const MEDIUM_PENALTY = 5 ;
const SMALLEST_PENALTY = 1 ;

CGraphDrawingPheno::CGraphDrawingPheno(CGAPDriver &driver, int width,
                                          int height)
{
    m_Driver = driver ;
    m_Width = width ;
    m_Height = height ;
    m_pGrid = new CWordMatrix(height,width,EMPTY_CELL) ;
    m_GridIndex[0] = new int [m_Driver.GetNumNodes()] ;
    m_GridIndex[1] = new int [m_Driver.GetNumNodes()] ;
}

CGraphDrawingPheno::~CGraphDrawingPheno()
{
    delete m_pGrid ;
    delete [] m_GridIndex[0] ;
    delete [] m_GridIndex[1] ;
}

double CGraphDrawingPheno::CalcFitness()
{
    WORD numNodes = (WORD) m_Driver.GetNumNodes() ;
    long maxDist = (m_Width * m_Height) ;
    maxDist *= maxDist ;
    //set base fitness so even the worst case phenotype
    // will not bring fitness below 0
    int connectivity = m_Driver.GetConnectivity() ;
    double base_fitness = numNodes*(numNodes-1) * maxDist ; //connectivity;

    double fitness = base_fitness ;
    for (WORD node1=0; node1<numNodes; node1++) {
        int node1Connections = Max(m_Driver.GetNumConnections(node1),1);
        for (WORD node2=0; node2<numNodes; node2++) {
            if (node1 == node2)
                continue ;
            BOOL bConnected = m_Driver.Connected(node1,node2) ;
            int node2Connections = Max(m_Driver.GetNumConnections(node2),1);
            double distance = Distance(node1,node2) ;
            distance *= distance ;
            if (bConnected && distance > 4) {
                fitness -= distance ; //((node1Connections*node2Connections) ;
                continue ;
            }
            if (!bConnected && distance <= 4) {
                fitness -= 4/distance ; //((node1Connections*node2Connections) ;
                continue ;
            }
        }
    }
    ASSERT(fitness >= 0);
    return fitness ;
}
```

```
}
void CGraphDrawingPheno::Decode(PTGenotype pGeno)
{
    WORD numNodes = (WORD) m_Driver.GetNumNodes() ;
    int rowAlleleLen = m_Driver.CalcRowAlleleLength() ;
    int colAlleleLen = m_Driver.CalcColAlleleLength() ;
    int offset = 0 ;
    for (WORD node=0; node<numNodes; node++) {
        char rowAllele[16], colAllele[16] ;
        //we know that these are no bigger than sizeof(WORD)
        for (int bit=0; bit<rowAlleleLen; bit++)
            rowAllele[bit] = pGeno->GetExpressedGeneValue(offset++,0) ;
        for (bit=0; bit<colAlleleLen; bit++)
            colAllele[bit] = pGeno->GetExpressedGeneValue(offset++,0) ;
        int codedRow = AllelesToInt(rowAllele,0, rowAlleleLen-1) ;
        int codedCol = AllelesToInt(colAllele,0, colAlleleLen-1) ;
        int actualRow, actualCol ;
        GetNearestEmptyCell(codedRow,codedCol,actualRow,actualCol) ;
        m_pGrid->SetAt(actualRow, actualCol, node) ;
        m_GridIndex[0][node] = actualRow ;
        m_GridIndex[1][node] = actualCol ;
    }
}

PTPhenotype CGraphDrawingPheno::Copy()
{
    CGraphDrawingPheno * pPheno =
        new CGraphDrawingPheno(m_Driver,m_Height,m_Width) ;
    return pPheno ;
    //don't copy values because these are derived by the genotype via Decode
}

void CGraphDrawingPheno::GetPhenoInfo(void *pInfoStruct)
{
    *((CWordMatrix **)pInfoStruct) = m_pGrid ;
}

//Algorithm resolves collisions by searching around the neighborhood of
// (row,col) in the grid for an empty cell. The row and col of the empty cell
// is returned in actualRow and actualCol.
void CGraphDrawingPheno::GetNearestEmptyCell(const int row, const int col,
                                              int &actualRow, int &actualCol)
{
    //insure we are in range!
    actualRow = row % m_Height ;
    actualCol = col % m_Width ;
    //if we find an empty cell then no search necessary
    if (m_pGrid->GetAt(actualRow,actualCol) == EMPTY_CELL)
        return ;
    else { //search for "nearest" empty cell
        int maxDist = Max(m_Height,m_Width) ;
        int actualRow2 = actualRow ; //save actuals
        int actualCol2 = actualCol ;
        //start at a distance of 1 and search outward
        for (int dist=1; dist<maxDist; dist++) {
            //First check "sides"
            for (int i=-dist; i<=dist; i++) {
                for (int j=-dist; j<=dist; j++) {
                    if (i!=j && (j==dist || j==--dist || i==dist || i==--dist)) {
                        actualCol = actualCol2+j ;
                        actualRow = actualRow2+i ;
                        if (actualCol >= 0 && actualCol
                            < m_Width &&
                            actualRow >= 0 && actualRow
                            < m_Height &&
                            m_pGrid->GetAt(actualRow,actualCol) == EMPTY_CELL)
                            return ;
                    } //for j
                } //for i
            } //Now check 4 corner cells
            actualCol = actualCol2+dist ;
            actualRow = actualRow2+dist ;
            if (actualCol < m_Width &&
                actualRow < m_Height &&
                m_pGrid->GetAt(actualRow,actualCol) ==
                    EMPTY_CELL)
                return ;
            actualCol = actualCol2-dist ;
            actualRow = actualRow2-dist ;
            if (actualCol >= 0 &&
                actualRow < m_Height &&
                m_pGrid->GetAt(actualRow,actualCol) ==
                    EMPTY_CELL)
                return ;
            actualCol = actualCol2+dist ;
            actualRow = actualRow2-dist ;
            if (actualCol < m_Width &&
                actualRow >= 0 &&
                m_pGrid->GetAt(actualRow,actualCol) ==
                    EMPTY_CELL)
                return ;
            actualCol = actualCol2-dist ;
            actualRow = actualRow2-dist ;
            if (actualCol >= 0 &&
                actualRow >= 0 &&
                m_pGrid->GetAt(actualRow,actualCol) ==
                    EMPTY_CELL)
                return ;
        } //for dist
    } //else
    return ;
}

//Return TRUE if node1 is adjacent to node2 on the grid
BOOL CGraphDrawingPheno::Adjacent(WORD node1, WORD node2)
{
    int row1, col1 ;
    if (!FindNode(node1,row1,col1))
        return FALSE ;
    int row2, col2 ;
    //look up
}
```



```

row2=row1-1 ;
if (row2 >= 0 && m_pGrid->GetAt(row2,col1) == node2)
    return TRUE ;
//look down
row2=row1+1 ;
if (row2 < m_Height && m_pGrid->GetAt(row2,col1) == node2)
    return TRUE ;
//look left
col2=col1-1 ;
if (col2 >= 0 && m_pGrid->GetAt(row1,col2) == node2)
    return TRUE ;
//look right
col2=col1+1 ;
if (col2 < m_Width && m_pGrid->GetAt(row1,col2) == node2)
    return TRUE ;
return FALSE ;
}
//Return TRUE if node1 is diagonal to node2 on the grid
BOOL CGraphDrawingPheno::Diagonal(WORD node1, WORD node2)
{
    int row1, col1 ;
    if (!FindNode(node1,row1,col1))
        return FALSE ;
    int row2, col2 ;
    //look upper left
    row2=row1-1 ;
    col2=col1-1 ;
    if (row2 >= 0 && col2 >= 0 && m_pGrid->GetAt(row2,col2) == node2)
        return TRUE ;
    //look lower left
    row2=row1+1 ;
    col2=col1-1 ;
    if (row2 < m_Height && col2 >= 0 && m_pGrid->GetAt(row2,col1) == node2)
        return TRUE ;
    //look lower right
    row2=row1+1 ;
    col2=col1+1 ;
    if (row2 < m_Height && col2 < m_Width && m_pGrid->GetAt(row1,col2) ==
        node2)
        return TRUE ;
    //look upper right
    row2=row1-1 ;
    col2=col1+1 ;
    if (row2 >= 0 && col2 < m_Width && m_pGrid->GetAt(row1,col2) == node2)
        return TRUE ;
    return FALSE ;
}
//Return the Euclidean distance between nodes on the grid
double CGraphDrawingPheno::Distance(WORD node1, WORD node2)
{
    int row1, col1, row2, col2 ;
    if (FindNode(node1,row1,col1) && FindNode(node2,row2,col2)) {
        double diffRow = row1 - row2 ;
        double diffCol = col1 - col2 ;
        return sqrt(diffRow*diffRow + diffCol*diffCol) ;
    }
    else
        return sqrt(m_Height*m_Height + m_Width*m_Width) ;
}
//Return the recti-linear distance between nodes on the grid
double CGraphDrawingPheno::RectDistance(WORD node1, WORD node2)
{
    int row1, col1, row2, col2 ;
    if (FindNode(node1,row1,col1) && FindNode(node2,row2,col2)) {
        double diffRow = row1 - row2 ;
        double diffCol = col1 - col2 ;
        return Abs(diffRow) + Abs(diffCol) ;
    }
    else
        return m_Height + m_Width ; //really an error ???
}
//Use an index to quickly locate a node on the grid
BOOL CGraphDrawingPheno::FindNode(const WORD node, int &row, int &col)
{
    if (node >= m_Driver.GetNumNodes())
        return FALSE ;
    row = m_GridIndex[0][node] ;
    col = m_GridIndex[1][node] ;
    return TRUE ;
}

```

Listing Three

```

//File: GDRIVER.H
#ifndef __GDRIVER_H__
#define __GDRIVER_H__
//flag an empty cell in the grid
const EMPTY_CELL = 0xFFFF ;

class CGAGraphDriver
{
//Interface
public:
    CGAGraphDriver(int numNodes, int width, int height) ;
    ~CGAGraphDriver() ;
    void SetGraph(CWordMatrix &graph) ;
    void Optimize(int numGenerations) ;
    void DrawOptimized(CDC &dc) ;
    void DrawUnOptimized(CDC &dc) ;
    //Query members (const)
    //Calc the length of a chromosome
    //needed based on the graph and grid
    UINT CalcChromosomeLength() const ;
    UINT CalcRowAlleleLength() const ;
    UINT CalcColAlleleLength() const ;
    int GetWidth() const ;
    int GetHeight() const ;
    int GetNumNodes() const ;
    BOOL Connected(WORD node1, WORD node2) const ;
    int GetNumConnections(WORD node) const ;
    int GetConnectivity() ;

```

```

    void Stop() ;
    PTIndividual m_pBest ;
    PTIndividual m_pWorst ;
    BOOL m_Stop ;
    //Implementation
private:
    //Draw the graph in this grid
    void Draw(CDC &dc, CWordMatrix &Grid) ;
    //num nodes in the graph
    int m_NumGraphNodes ;
    //width of grid to draw on (in cells)
    int m_GridWidth ;
    //height of grid to draw on (in cells)
    int m_GridHeight ;
    //connection table representation of a graph
    CWordMatrix m_pGraph ;
    //GA that will find the "optimal" drawing
    //of the graph on the grid
    TBasicGA m_pTheGA ;
} ;

```

Listing Four

```

//File: GDRIVER.CPP
//Used as an interface class to the GA.
//Stores the representation of the graph as
//a connection grid.

//required headers
#include "stdafx.h"

//Headers needed for EOS programs
//You need EOS v1.1 to compile this code
#include "eos.h"
#include "eosutil.h"
#include "geno.h"
#include "indiv.h"
#include "gaenviro.h"

//headers specific to graph GA
#include "wmatrix.h"
#include "gdriver.h"
#include "grphutil.h"
#include "graphga.h"

//GA parameters used, these need not be
//hard coded in advanced implementations
const int POP_SIZE = 20 ;
const double PX = 0.7 ;
const double PM = 0.03 ;
const double RAND_SEED=0.76451 ;

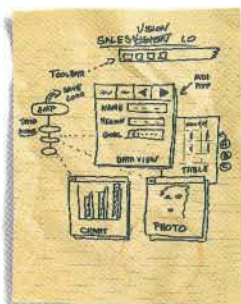
//DRAWING parameters used, these need not be
//hard coded in advanced implementations
const int CELL_WIDTH = 30 ;
const int CELL_HEIGHT = 30 ;
const int CELL_SPACE = 30 ;

//Driver constructor initializes a graph with numNodes and a
//grid that the graph will be optimized to draw on (width x height)
CGAGraphDriver::CGAGraphDriver(int numNodes, int width, int height)
{
    m_NumGraphNodes = numNodes ;
    m_GridWidth = width ;
    m_GridHeight = height ;
    //graph represented as boolean connection matrix
    m_pGraph = new CWordMatrix(m_NumGraphNodes,m_NumGraphNodes) ;
    //The Graph GA object
    m_pTheGA = new CGraphDriverGA(*this) ;
    m_pBest = NULL ;
    m_pWorst = NULL ;
    m_Stop = FALSE ;
}
//Clean up in the destructor
CGAGraphDriver::~CGAGraphDriver()
{
    delete m_pGraph ;
    delete m_pTheGA ;
}
//set the connections from graph into the member m_pGraph
void CGAGraphDriver::SetGraph(CWordMatrix &graph)
{
    for (int row = 0 ; row < m_NumGraphNodes; row++)
        for (int col = 0 ; col < m_NumGraphNodes; col++)
            m_pGraph->SetAt(row,col,graph[row][col]) ;
}
// Optimize the drawing of the graph by first initializing the GA's population
// and environment. Then execute the GA for numGenerations generations
void CGAGraphDriver::Optimize(int numGenerations)
{
    m_pTheGA->CreatePopulation(POP_SIZE) ;
    m_pTheGA->CreateEnvironment(PX,PM,RAND_SEED) ;
    m_pTheGA->Evolve(numGenerations) ;
}
//Draw the optimized graph on the Windows DC
void CGAGraphDriver::DrawOptimized(CDC &dc)
{
    CWordMatrix *pGrid ;
    m_pBest->GetPhenoInfo(&pGrid) ;
    Draw(dc,*pGrid) ;
}
//Draw the un-optimized graph on the Windows DC
void CGAGraphDriver::DrawUnOptimized(CDC &dc)
{
    CWordMatrix *pGrid ;
    m_pWorst->GetPhenoInfo(&pGrid) ;
    Draw(dc,*pGrid) ;
}

```

(continued on page 147)

Monday - The assignment



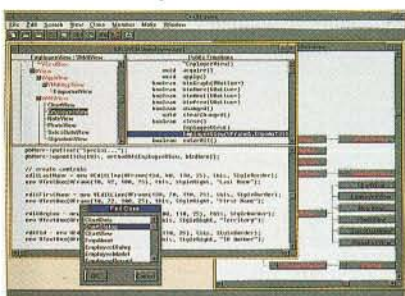
"OH, NO! I agreed to get this application built by Friday... on Windows, Motif and the Mac."

Tuesday - Use C++/Views™ visual interface builder



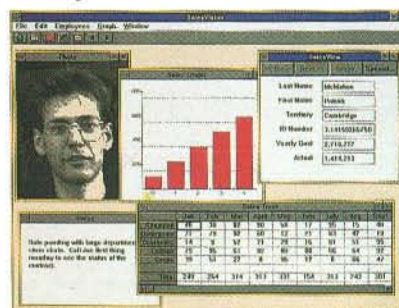
"Hey, I'm already ahead of schedule. I've got my dialogs laid out - I just have to finish the menus."

Wednesday - Use C++/Browse™



"Now, I'll use the class browser to create my classes and attach them to the dialogs and menus."

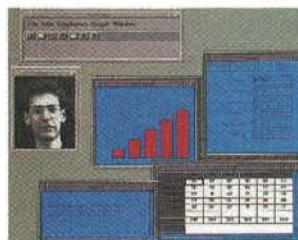
Thursday - Finish MS® Windows™ version



"Yess-ss!"

Friday - Porting frenzy

OS/Motif



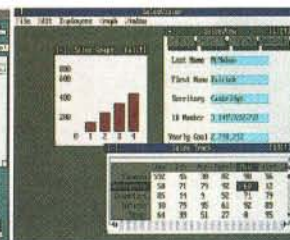
OS/2®



Macintosh®



DOS Text



"I had some extra time, so I put it on OS/2 and DOS too."

How to build applications on an insane schedule without going crazy.

Shock everyone (including yourself), and get your GUI development done quickly and portably. Simply use C++/Views 3.0, the best visual programming tool for multi-platform development.

C++/Views 3.0 includes a new visual interface builder with powerful features such as callback editing, geometry, and portable resource files.

Its browsing utility lets you create, derive, and edit C++ classes interactively. And it automatically keeps your source, header, and make files synchronized.

Its class library provides an comprehensive set of GUI, extended GUI, data, and file management classes. Best of all, C++/Views unique

design gives you object-oriented programming power and flexibility, without sacrificing platform-specific features, performance, or native look-and-feel.

Call us today for more information and a free white paper.

Call 800-237-1873. Fax 508-820-0035.

In Europe, call (44) 71-799-2434. Fax (44) 71-799-2552.

CIRCLE NO. 365 ON READER SERVICE CARD

LIANT



A Little RISC Lands Apple in the Soup

Michael Swaine

In the early 1980s, the British microcomputer market was dominated by British companies, primarily Sinclair and Acorn.

It was an unlikely scenario.

The microcomputer revolution was by this time becoming institutionalized. What, only a few years before, had been a marginal market of electronics hobbyists selling to other electronics hobbyists had become a venture-capital-attracting international industry. IBM had come in and legitimized the industry, was the commonly heard—and true, even if incomplete—explanation.

All the early shots in this revolution had been fired in the United States, and all the big companies—no surprise—were U.S. companies, some of which had established manufacturing facilities in Europe. The European market, taken as a whole, was only a fraction of the U.S. market. The British market was a fraction of that fraction, and, unlike some European countries, Britain didn't have high tariffs to keep out American computers. By all logic, American computer companies should have been able to walk all over the homegrown brands.

But that's not what happened. British computer companies were bucking the odds and winning. What was going on?

Who Were These Guys?

One of the things that stands out when you look at the British microcomputer scene in those days is the Cambridge connection. Sinclair and Acorn had Cambridge University connections in common, and Acorn in particular maintained close ties with the university, drawing on it for personnel, ideas, and support. Cambridge may have been one strength of these companies.

But Sinclair and Acorn differed in many ways. For one thing, Clive Sinclair went for the high-concept products: The World's Cheapest Computer, The First Practical Electric Car. The Acorn crew were less flamboyant. They just built a computer.

The Sinclair computer was one of the first users of the Zilog Z80, arguably the first microprocessor created specifically

to be the CPU of a personal computer. Arguably. The Acorn used a chip originally intended for controller use: the Rockwell 6502. The Acorn developers got to be experts in the 6502, just as Apple cofounder Steve Wozniak did.

Clive Sinclair, like Nolan Bushnell in the United States, founded several companies, explored diverse industries, and had flashes of high visibility; Sinclair, though, has been off American radar for years. The Acorn team prospered with less abrupt ups and downs and has significant visibility today. It was the BBC deal that made their fortune.

The British Broadcasting Company had decided to launch a computer-education television show that would run throughout the UK, and it wanted a BBC microcomputer to sell to viewers of the show. It was a savvy plan, and when Acorn got the BBC contract, both Acorn and the BBC thought that they could sell over ten thousand computers despite the small size of the nascent British market.

To date, Acorn has sold nearly two million BBC Micro-compatibles, and the company has grown from a typical microcomputer company of the early '80s with a staff of a couple dozen to a multi-million-pound company with hundreds of employees.

When it came time, in the mid-1980s, to admit that the 6502 had had its day, the Acorn guys did something telling. Rather than accept the conventional wisdom about the "right" microprocessor for the next generation of computers, they fell back on their expertise, or perhaps just their old habits. They designed their own.

What they came up with was the kind of chip you might expect old 6502 hackers to design: a small instruction set, low power consumption, small die size, potentially low cost. It may have been of only academic interest to them that these are now the characteristics of low-end RISC chips. They weren't trying to develop the first commercial RISC processor. They just wanted a better 6502.

What they came up with was the Acorn RISC Machine, or ARM. The first

ARM chip was shown fully functional in April of 1985. It operated reliably at 8 MHz, although designed to operate with a 4-MHz clock. It was a 3 μ device of about 25,000 transistors. Initially, the ARM1 was offered as a coprocessor in the BBC computer. The second generation ARM2 was used by Radius in one of its first graphics accelerator cards for the Macintosh. The ARM2 also saw service in the movies, being used in the robotic controller from MicroRobotics of Cambridge, England, that controlled the robot turtles in the movie *Teenage Mutant Ninja Turtles*.

Meanwhile, Back in the Colonies...

Apple formed its Advanced Technology Group (ATG) in 1986. At that time Acorn, facing competitive pressures from clones, had just been acquired by Olivetti and was soon to release its first ARM-based computer, the Archimedes, to a lukewarm response. Apple's ATG was chartered to explore new technologies that could be of use to Apple in the '90s. One technology that ATG evaluated and took note of for possible inclusion in Apple products was Acorn's ARM processor, but nothing was done with the ARM at the time.

Somewhat later, a skunkworks within ATG called the Advanced Products Group (APG) took on the mission of developing a new system architecture that they were calling Newton. The trip to Newton had a lot of side trips and blind alleys. It was apparently Michael Chao's Knowledge Navigator pitch to John Sculley that tipped the balance from a tablet form factor to the handheld device that Apple eventually released.

One of the other alleys explored involved the microprocessor. For some time the AT&T Hobbit chip was considered. What they were looking for was a processor with characteristics that sounded like those of a microcontroller rather than a computer CPU: small die size, low cost, low power consumption, instruction set efficiency, ease of embedding in ASIC designs. In 1990, RISC

(continued on page 112)

{ setBookstore="barnes & for (selection>none_are=)

OPERATING SYSTEMS

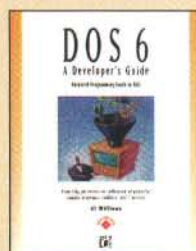


Windows 3.1: A Developer's Guide, 2nd Edition

by Jeffrey M. Richter

This highly regarded best-seller has been updated and revised to cover Windows 3.1. Covers new features, including new Windows 3.1 hooks, subclassing and super-classing windows. Packed with valuable illustrations, utilities and source-code examples.

Disk contains 12 complete applications. (BK/DISK)
List Price \$39.95

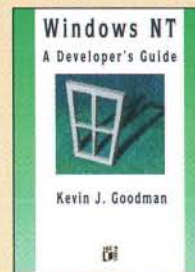
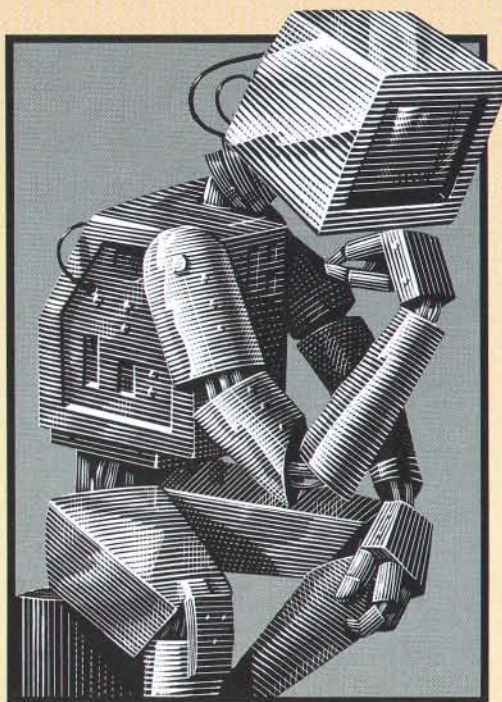


DOS 6: A Developer's Guide

by Al Williams

This bestseller has been updated to cover the latest DOS release. It provides a solid understanding of the DOS operating environment plus full coverage of its newest features. You'll find new information on interacting between DOS and

Windows, CD-ROM access, Vesa SuperVGA techniques and XMS. The disk is loaded with sample programs, toolkits and libraries. (BK/DISK)
List Price \$39.95



Windows NT: A Developer's Guide

by Kevin J. Goodman

Build your 32-bit programming skills quickly with this guide to Windows NT, the new 32-bit operating system from Microsoft. Filled with thorough discussions and examples, it covers the ins and outs of Windows NT programming, teaching you how to develop Win32 and

Win32 applications. Disk includes sample code. (BK/DISK)
List Price \$39.95

ALSO AVAILABLE

teach yourself... Windows 3.1

by Al Stevens

List Price \$21.95

teach yourself... DOS 6

by Al Stevens

List Price \$21.95

teach yourself... OS/2 2.1

by Judi N. Fernandez

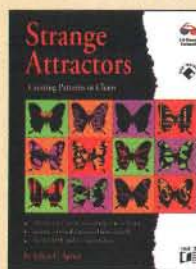
List Price \$21.95

teach yourself... Windows NT

by Hayagriva Rao

List Price \$21.95

PROGRAMMING CLASSICS

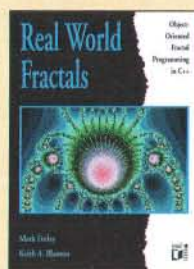


Strange Attractors

by Julien C. Sprott

Learn to create spectacular fractal images with this complete book/disk package. Step-by-step instructions lead you through the creation of a program that produces an endless number of patterns and musical sounds. It contains over 350 examples of computer art plus a color insert, an interactive disk

and 3-D glasses. (BK/DISK)
List Price \$39.95

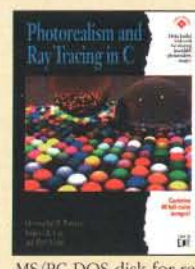


Real World Fractals

by Mark Finlay
and Keith A. Blanton

Learn to create exciting fractals using object-oriented programming techniques. This book/disk package explores the latest advances in fractal modeling, showing you how to apply the techniques to real-life applications. Contains eight pages of full-color fractals and a disk with complete source code. (BK/DISK)

List Price \$39.95

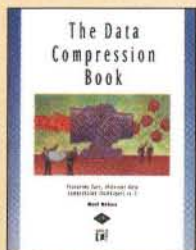


Photorealism and Ray Tracing in C

by Christopher Watkins,
Stephen Coy and Mark Finlay

This book puts the tools in your hands to produce photorealistic, 3-D images on PCs. Includes a section on ray tracing, plus tips for producing sample images as well as creating original designs. Source code on

MS/PC-DOS disk for reproducing and customizing sample images. (BK/DISK)
List Price \$44.95

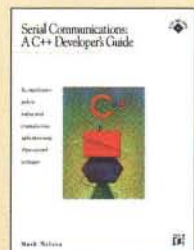


The Data Compression Book

by Mark Nelson

An authoritative guide for advanced C programmers. Details various data compression techniques, explaining the theory behind each and showing how to apply them to significantly increase your system's storage capacity. MS/PC-DOS disk

contains sample source code. (BK/DISK)
List Price \$39.95

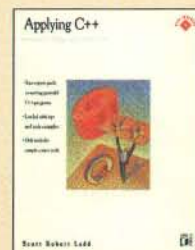


Serial Communications: A C++ Developer's Guide

by Mark Nelson

A hands-on guide to mastering object-oriented techniques in writing software for modems, BBSS and other communications systems, this book covers the latest C compilers from Microsoft, Borland and Zortech. (BK/DISK)

List Price \$44.95



Applying C++

by Scott Robert Ladd

Intermediate level programmers...this is your next book! Learn how to design and maintain clean, efficient C++ applications and do it by using the very tricks, techniques and strategies of the industry's acknowledged C++ gurus. Disk includes complete source code. (BK/DISK)

List Price \$34.95

noble bookstar bookstop"; get next book There(); }

WINDOWS PROGRAMMING

Writing Windows Applications from Start to Finish



Writing Windows Applications from Start to Finish

by Dave Edson

Using a real-world example, this book guides programmers through the steps involved in writing Windows applications, including user interface design, file content, data structures, coding, testing and documentation. Readers not only learn how to plan and design an application from start to finish but also acquire a home inventory program. Disk contains complete source code. (BK/DISK)

List Price \$39.95

Windows Programming with Borland C++



Windows Programming with Borland C++

by Steve Oualline

Learn to create Windows applications using Borland C++ 3.1. This hands-on guide shows you how. It leads you through the design, implementation and debugging process used to create a program. You'll learn how to create various Windows applications including a Find File program and a fish screen saver. Complete with source code disk. (BK/DISK)

List Price \$39.95

BORLAND PASCAL WITH OBJECTS 7.0

Borland Pascal with Objects 7.0

by José de Jesús and Abad Godoy

Borland Pascal with Objects 7.0 integrates DOS and Windows like no other development tool on the market. This book will teach you object-oriented programming and how to incorporate existing objects into your Pascal programs. Structured as a tutorial, this book will guide you from the basics of Pascal for DOS, through Turbo Vision and up to ObjectWindows. If you're a beginner, you can use this book to build your expertise from the bottom up. If you're an experienced programmer, you can leaf through familiar areas into more advanced topics. (BK/DISK)

List Price \$39.95

Plug & Play Programming

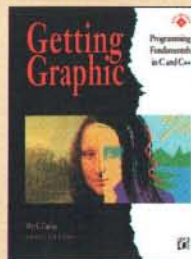


Plug & Play Programming

by William Wong

This book introduces a new programming technique based on an object-oriented class called "plugs." It teaches programmers how to use plugs to design programs with interchangeable components that can easily be ported from one program to another without modification. Includes MS/PC DOS disk containing ready-to-use libraries of plugs plus source code. (BK/DISK)

List Price \$39.95

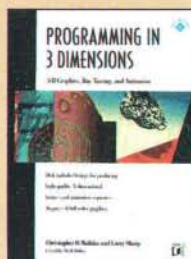


Getting Graphic: Programming Fundamentals in C and C++

by Mark Finlay

This book teaches the fundamentals of graphics programming. It shows C and C++ programmers how to plot points on a screen, draw geometric shapes, design 3-D figures and more. This book/disk package is filled with sophisticated and usable source code examples and sample graphic images. *Getting Graphic* is a perfect introduction to the exciting world of graphics. (BK/DISK)

List Price \$39.95

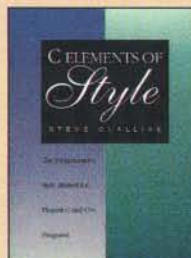


Programming in 3 Dimensions

by Christopher D. Watkins and Larry Sharp

Required reading! This one is for all computer graphics enthusiasts who want a detailed look at 3-D graphics and modeling. Also features discussions of popular ray tracing methods and computer animation. Includes eight pages of full-color graphics. Provides C source code and numerous examples. MS/PC-DOS disk contains sample source code. A must have! (BK/DISK)

List Price \$39.95



C Elements of Style

by Steve Oualline

This concise guide covers the rules of good program design, teaching C and C++ programmers how to write programs that can be easily read, understood and maintained by others. Whether you're a student or professional programmer, you'll benefit from the many tips and techniques for constructing elegant, reliable code.

List Price \$21.95



C++ Database Development

by Al Stevens

Al Stevens, the popular *Dr. Dobbs* columnist, provides all the tools you need to explore the full potential of the C++ object-oriented language. Learn to use C++ to design and develop utilities and database management programs. Topics include database fundamentals and design, database management, building the software and much more.

List Price \$24.95

Call 1-800-344-2470 for the location nearest you.

Barnes & Noble | ★ BOOKSTAR | BOOKSTOP

(continued from page 109)

looked promising, and ARM looked particularly good.

To ensure that future ARM processors would fit Apple's evolving needs, Apple made a deal. It was an early example of the joint ventures that Apple continues to pursue today. Apple UK joined forces with Acorn and VLSI Technology, with whom Acorn had worked in producing the first ARM chips, to form ARM Ltd.

ARM Ltd.'s ARM 610 became the processor for the first Newton devices, the Apple MessagePad and Sharp ExpertPad. (ARM6 devices like the ARM 610 really represent the fourth generation of ARM devices; apparently the numbering skipped 4 and 5.)

ARM was on a roll. In 1992, 3DO announced that the ARM60 would be used in its Interactive Multiplayer. ARM6 devices are also seeing use in controller applications, such as fuzzy-logic controllers.

The ARM6 family embodies full 32-bit addressing and support for both Big-endianness and Little-endianness, a requirement imposed by Apple. The ARM610 includes a 4-Kbyte cache, a write buffer, and a MMU, all in a package smaller than a 386. The MMU implements memory domains and permissions designed to provide hardware support for modern operating-system memory-management strategies like multilevel memory protection, memory paging, demand-paged virtual memory, and object-oriented memory with background garbage collection. The last of these turns out to be crucial to the Newton model for object storage.

The rest of this column looks at some of the characteristics of that model.

A Little Selfishness

Newton's model of object-oriented technology is reported to be related to SELF, an object-oriented dynamic language developed by Smith and Unger at Stanford University about the time the ARM1 chip was seeing first silicon. NewtonScript is not SELF, though, or Dylan, or any other language. It has some unique characteristics.

One characteristic that NewtonScript does share with SELF is the "everything is an object" approach. The SELF model is unusual among object-oriented languages in that it isn't built around classes. The slogan "everything is an object" means that objects inherit directly from other "prototype" objects, as distinct from the more familiar class-based inheritance.

Newton's object-oriented language, NewtonScript, diverges from SELF in many ways, but has much the same

spirit. It has prototype inheritance, as well as "parent" inheritance. But not everything is an object to NewtonScript. Chunks of data that can fit into 32 bits (integers, characters, Boolean values) are addressed via immediate reference, while everything else is a pointer reference. All these pointer-referenced data are stored in the heap as, yes, objects. Some object-data types are: symbols, reals, arrays, strings, and frames. The most important type of object in the Newton object-storage model is the frame.

A frame is a data structure containing named references to objects of arbitrary data type. It's much like a *struct* or record in other languages. A frame can also contain functions.

Example 1 is a typical NewtonScript frame. Frames in NewtonScript are delimited by braces ({}). The named data items within a frame are called "slots." Each slot is specified by its name, a colon, and its value. The slots are separated from one another by commas. Example 1 shows a *_proto* slot (more about this shortly), an integer constant slot, a Boolean constant slot, a string constant slot, a function slot (this is how methods are implemented in NewtonScript), and a slot that is itself a frame.

The *_proto* slot indicates one of the modes of inheritance, prototype inheritance. To establish that frame 2 inherits in this way from frame 1, you give frame 2 a *_proto* slot and give that slot a reference to frame 1 as its value. Frame 1 is then frame 2's prototype. Frame 2 can use (inherit) slots of frame 1, can override them with its own slot declarations, and can have additional slots that frame 1 doesn't have. Since functions can appear in frame slots, functions can also be overridden and inherited in this same way.

By the way, to send that method *exampleFunction* as a message to the frame *exampleFrame*, the syntax is *exampleFrame : exampleFunction*.

A couple of points will indicate how you work with this kind of inheritance: Inheritance is by reference, and prototypes can be in ROM. The implication

```
exampleFrame := {
  _proto: protoFrame,
  index: 1,
  active: TRUE,
  name: "Name of Frame",
  exampleFunction:
    func(param)
    begin
      return param * 10;
    end
  otherFrame:
    {
      owner: "Mike Swaine",
      ownerAddress: "72511,172"
    }
}
```

Example 1: A NewtonScript frame.

is that if there is any chance that a reference to a certain slot may be a reference to ROM, you should declare that slot in frame 2, even though it is declared in frame 1 and inherited from it.

In fact, the whole Newton user interface essentially resides in prototypes in ROM, and you can use them as the prototypes for components of your applications. Simple Newton applications can be developed without any actual coding by using visual programming tools in the Newton Toolkit (NTK). These tools mainly facilitate this process of using ROM prototypes as the prototypes for components of your application. More complex applications will require some actual coding, of course, and it should be noted that only the user-interface elements can be used in this way. The rest of your app has to be built the hard way.

Look for the Union Label

To understand how Newton stores object data, you need to know about stores, soups, and entries.

Newton objects can, at least for the current devices, reside in one of two places: in memory (ROM or RAM) or on a PCMCIA card. The memory and the card are called "stores." Other stores may be available on future Newton devices.

Stores contain collections of data called "soups." All the data in a store are in soups, and a store can hold many soups. If a store is like a volume, a soup is like a database on the volume.

Soups are made up of "entries." An entry is a frame. If a soup is like a database, an entry is like a record.

This model—physical stores containing soups made up of entries, and entries that are *struct*-like frames of object data—shows that Newton objects basically reside on Newton's physical storage devices, but it creates a false impression.

Because it isn't the simple soups that matter most in Newton software development, but cross-store collections called "union soups." Union soups seamlessly merge data from soups of different stores. If programmers use union soups rather than soups, then users can always decide where they want their data stored. In a machine with less than 200K of user-available RAM, you can be sure that's an issue. The moral for Newton developers: Use union soups.

Naturally, there's an exception to this rule. Preferences are stored in the System soup in ROM only. Every application adds at least one entry to this soup, which is not a union soup.

All existing soups (the "names" soup used by the bundled Names application,

for example) are available to your application, and you are encouraged to use them. You can add your own data to these existing soups by adding a slot. To avoid conflicts, Apple encourages you to add just one slot, using your *app-Symbol* as the name entry for the slot.

Note the distinction: Adding an entry to a soup is like adding a record to a database. Adding a slot is like adding a field.

Soup Management

Besides automatic garbage collection, Newton provides a lot of built-in data management. Soups automatically maintain indexes of their entries. You specify these indexes when you create a soup, but indexes can be added and removed dynamically. Currently, the only kind of index supported is "slot," but future versions of NewtonScript may support others. Using a slot index means that the index key is the value of a particular slot that appears in each entry.

The function *theStore : createSoup (soupNameString, indexArray)* creates a soup of the specified name in the store named *theStore*. *IndexArray* is a frame describing the initial index(es) you are creating for the store. You don't have to create any, since indexes can be added later. Soups can contain any mishmash of entries, but unless all entries have at least one slot in common, it won't be possible to specify an index that lets you search the whole soup. Some points on managing soup entries: When you add an entry to a soup, you actually add the transitive closure of the entry. Altering an entry doesn't update the store; you need to call *EntryChange*. The Newton operating system calls *EntryChange* every so often when idle, but applications will typically have to know when to call *EntryChange* themselves. The only way to get at the entries in a soup is via a "query." A query can use an index, or some other kind of search, like searching all string slots in all entries for a specified search string. A query returns a set of entries, and these entries are then accessed through an object called a "cursor."

A cursor is a pointer to one of the entries in this returned set. The cursor is advanced to the next entry in the set or otherwise repositioned by sending it messages.

The Newton approach to handling persistent-object data has some distinctive and, I think, interesting characteristics. I suspect I'll have more to say about it in future columns.

DDJ

To vote for your favorite article, circle inquiry no. 11.

Real UNIX SVR4 Tools On Windows NT



SPECIAL OFFER

If you know UNIX and are using NT, this product can pay for itself in a week. If you're porting UNIX code to NT, it could pay for itself the first day. We used Portage development tools and libraries to port over 120 UNIX utilities to NT from SVR4 source code, including both *ksh* and *csh*.

For a limited time, you can get the SVR4 base system for only \$395, development tools for \$495, or save hundreds of dollars and order both for only \$695! Call or fax us for complete product literature, or to place your order. Prices on Alpha and MIPS are \$100 higher. Call today and become much more productive on NT (ask about imminent availability on "Chicago").

Portage™ Features

- A direct port of the UNIX® SVR4 source code to Windows NT™. All necessary UNIX system calls are supported.
- Over 120 UNIX SVR4 commands available on Windows NT (Intel, Alpha, & MIPS versions) --- *vi*, *ls*, *grep*, *awk*, *sed*, *ed*, *diff*, etc.
- Run UNIX commands from the NT shell, and NT commands from the UNIX shell (both *ksh* and *csh*).
- Complete on-line manual pages as Windows Help.
- Development tools include *yacc*, *lex*, *make*, *SCCS*, ..., along with our UNIX system call and subroutine libraries for NT.

CONSENSYS

1-800-388-1896

Phone: 1-905-940-2900

Fax: 1-905-940-2903

Trademarks/Owner: Consensus, Portage/Consensus Corp.; UNIX/UNIX System Laboratories Inc.; Windows NT/Microsoft Corp.

CIRCLE NO. 683 ON READER SERVICE CARD

Now the *EASIEST* Way to Add Video Compression to Your Applications...

Is Also the Least Expensive!

JPEG

MPEG

Px64

VDS™ Kit
Video Decompression Source

You already know the value of compressed video for adding new dimensions of excitement to your applications. Yet, until now such compression and decompression products have been at a staggering cost per copy, if they've been available at all. But now Performance Computing is offering the same high quality video codec utilities at the highly compressed price of only \$249 suggested retail. With these ANSI-standard software MPEG, JPEG, and Px64 libraries, you can have the power of video for the price of a disk utility!

And the VDS Kits are affordable for OEMs and developers who want to sell video based applications. Our royalty free licenses make it as easy to do business as our VDS Kits make it easy to develop multimedia applications in the first place. And who else supports laptops, 7 operating systems, plus 20 CPU's?

So, if you're building a videoteleconference, retail photo kiosk, videophone, on-line training, or similar applications, you owe it to yourself to call us first.

(800) PCI-VIDEO
or fax: **(503) 297-0878**

VISA/MC/AMEX Accepted. Dealer Inquiries Invited.
1815 S.W. Marlow Ave., Suite 206
Portland, Oregon 97225



CIRCLE NO. 874 ON READER SERVICE CARD



Essential Chart for Windows

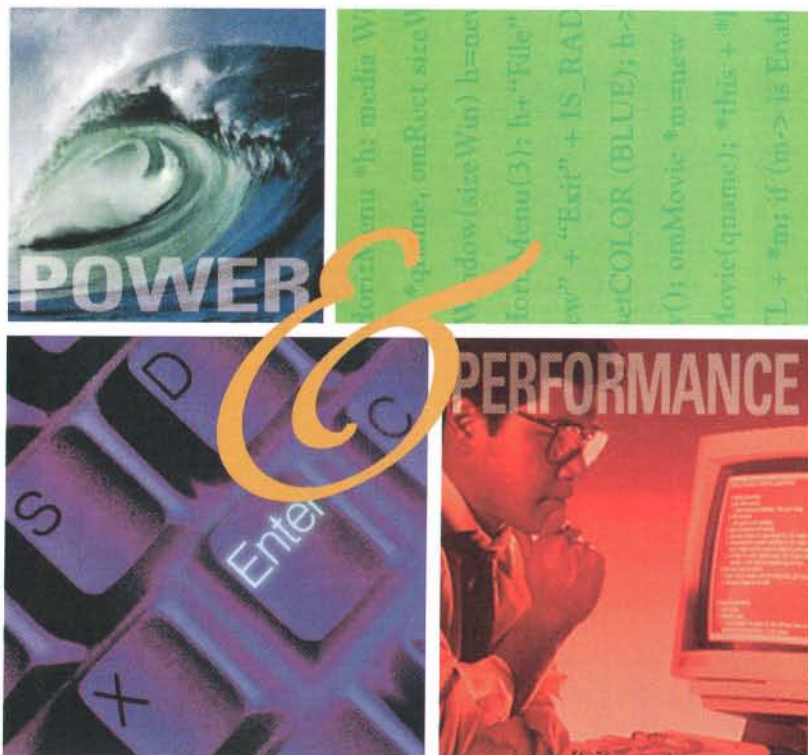
The most powerful and easiest way to develop charts for Windows

A graphical presentation of data adds important appeal and utility to any application. Essential Chart makes your data come alive in full color 2D or 3D, static or real-time charts and graphs. This comprehensive Windows DLL adds extensive functionality including true 3D rotation and perspective, and the ability to scale text using built-in True-Type font support. Plus, a powerful, time-saving CASE tool allows you to interactively create and design charts and chart templates without programming.

Features:

- 18 different chart types
- 2D and 3D bar, pie, pyramid, ribbon and area charts
- Financial and analysis chart types
- True 3D perspectives and rotations
- Real-time charting with live data
- Supports True-Type fonts for text scaling and resizing
- Windows printer support
- Displays chart in a parent or child window
- Source available

Pricing: \$399



object-Menu

The cross-platform application framework for C++

A good interface design can be the deciding factor in today's competitive marketplace; object-Menu makes it happen in several ways. Built-in aesthetics make it easy to create interfaced styling such as Windows, Motif, or your own custom design. Flexible interface configuration allows you to mold the interface to the application, not the other way around. And, portability to DOS, Windows/NT and OS/2 enable you to offer your product to multiple target markets with a single engineering effort. Additionally, object-Menu's intuitive architecture, straightforward methodology and Visual Design tool actually speed GUI development to allow more time to focus on your application.

Features:

- Currently supports DOS graphics, Microsoft Windows, Windows NT and OS/2.
- Visual Design Tool included with automatic code generation
- Extensive "behind the scenes support" keeps your programs small and easier to debug and maintain
- Comprehensive set of interface and graphics objects including multi-media objects
- Supports multiple browsers or child windows off the main application
- Extensive data entry capability including real time field formatting and validation as well as unlimited data types
- Hypertext help system with help authoring plus hypertext primitives
- Intermix graphical icons with menus, buttons, text
- Includes the MetaWindow/XL high performance graphics library
- Supports major C++ compilers and all popular DOS graphics libraries

Pricing:

Single platform (choose DOS, Windows, NT or OS/2)	\$299
Single platform with source	\$449
object-Menu Professional (DOS, Windows/NT with source)	\$699
object-Menu Professional-OS/2 (DOS, Windows/NT, OS/2 with source)	\$899

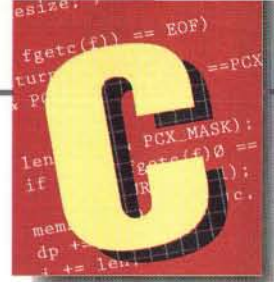
LIFEBOAT®
PUBLISHING

1163 Shrewsbury Ave. • Shrewsbury, NJ 07702
201-762-6965 • Fax 908-389-9227 • BBS 201-762-0339

CALL to ORDER
800-447-1955

Borland Nonsense: Ready, Aim, Shoot!

Al Stevens



The marksman: Borland. The target: Borland's foot. The weapon: Borland C++ 4.0's No-Nonsense License Statement.

Read the saga of how a company, known far and wide as the software developer's friend, dropped their guard, let their lawyers rewrite their no-nonsense license statement, and plugged themselves squarely in the pedal extremity.

Our story begins with the patent insanity. Unbeknownst to us, Borland holds a patent on their VROOMM overlay technology, and they have several other software patents pending. Those patents, when granted, will cover algorithms that are implemented within their libraries, DLLs, database engines, and other redistributable modules. In theory, when you build a program with their compiler, the executable code will contain algorithms covered by a Borland patent. Setting aside the question of the validity of software patents in general, the result is that you are distributing a program made with patented components. By law, you need a license from Borland to distribute those components.

Licenses can be obtained in many ways. You can pay a one-time fee for an unlimited license. You can pay a per-copy royalty. The holder can give you a royalty-free license. You can exchange patent licenses. Or you can be denied the license. If the patent holder does not want any competition, or does not want you in business for some reason, they can refuse to grant you a license. You would need to find another way to write your program.

Traditionally, Borland and other compiler vendors include this grant in the license conditions with which you tacitly agree when you break the seal and use the product.

The Borland Dilemma

Prior to version 4.0, Borland's C++ no-nonsense license statement made no mention of patents. It granted to each registered user a license to distribute compiled programs without additional fees being charged. But someone at Borland saw something wrong with that. They reasoned that a major competitor

could use Borland technology to build competing tools and applications.

As the self-professed dominant vendor of tools and applications, Borland found itself facing an internal conflict of agendas. The languages division wants to provide software developers with the best software development technology. The applications folks want to maintain dominance in a marketplace where competitors can use those superior Borland tools.

As one Borland spokesman put it, Microsoft could buy one copy of Turbo C++ for \$99.00 and receive unlimited use of the patented VROOMM technology in applications that would then compete with Borland applications. Borland wanted to keep the competition from using its patented technology against it and continue at the same time to be responsive to the needs of its language customers. It was the old cliché about having your cake and eating it, too, which is what Borland tried to do. But what it came up with was met by an overwhelming firestorm of user reaction.

What lit the fire? Well, in times past, you could distribute as many copies of programs as you wanted. Under the terms of the new no-nonsense license statement, you could distribute only up to 10,000 copies per year of your Borland-compiled application. To distribute more copies than that, you would have to get Borland's permission. The reasoning behind this peculiar condition, as spokespeople explained later, is that only large competitors are likely to be selling more than 10,000 copies per year.

D-Flat Gets a License

I wanted to learn more, so I set out to get a royalty-free license to distribute more than 10,000 copies of D-Flat. I called Borland and asked for their OEM licensing department, which is what the no-nonsense license statement says I should do. The operator connected me with Karen Rogers. When I asked if this was the OEM no-nonsense licensing department, she hesitated, laughed, and asked what my call was about. Karen is in Corporate Affairs. I told her what

I needed, and she transferred me to John Smart, Borland's patent lawyer. I told him what I wanted, and he said no problem. When he got my name, company, and the name of D-Flat, he recognized it, knew I was from the press, and we had a congenial conversation about the situation.

Getting the license was easy. I have it now and may distribute D-Flat without restriction. But the disturbing part is to get this license, users, potential Borland competitors or not, had to tell Borland about the product. Open the books, so to speak.

The Shift Hits the Kahn

Programmers around the world read the 10,000-copy restriction and went ballistic. There are many venues for software distribution where the developer cannot account for numbers. One is shareware. Another is the distribution of royalty-free redistributables that you develop for other programmers to use to develop programs which they distribute. Such as D-Flat. Get's hairy, doesn't it? But for whatever reason, no one gives up freedoms without a fight, particularly when they are taken away in the small print. Programmers felt betrayed and said so, loudly and with some emotion. The Borland forums on CompuServe burned with their complaints. Many vowed to return the package for a refund. Most demanded an explanation.

Borland Responded

Borland reacted to the outcry by posting a Q&A dialog on CompuServe that was supposed to clear up the matter. They announced their intention to revise the no-nonsense license statement to remove some of the restrictions, but the wording of the Q&A was vaguer than the original no-nonsense license statement, and it was not clear how they would deal with the problem short of removing all of the restrictions.

The 10,000-copy restriction was silly at best. Borland's VROOMM patent is the only one it has, although others are pending. One wonders who Borland was trying to contain. The Q&A document stated publicly that the restrictions

are directed only at certain large, litigious competitors and that others had nothing to worry about. It said, "If you are not a litigious competitor, then the restriction doesn't apply to you." How does Borland know who is going to sue them? It seems to be saying, "If you sue me, I'm taking my license back."

Smart narrowed the number of litigious competitors to two and would not name them but said that one of them was suing Borland now. That would be Lotus.

Listen up, Philippe. Overlays ain't that hard to figure out. Lotus can hire some fast and loose programmers and do their own overlay manager quicker than you can snap-roll your Waco. It isn't worth all this bad public relations just to force them to do that.

Speculation follows. Could be someone at Borland heard that Lotus wrote everything in Turbo C and is heavily committed to some compiler implementation-dependent stuff. That would be the *coup de grace*. Rig your no-nonsense license statement so that a big competitor, one who just happens to be suing your eyes out, cannot upgrade to the next version of their principal development tool. This is the only scenario that I can come up with that even remotely explains Borland's changes in attitude about patents. Nonetheless, I wonder about the ethics and legality of a license that is publicly waived for everyone except certain competitors. Borland told us that there are only two targets, and it gave us enough information to guess who the large, litigious competitors are. End of speculation.

Other Restrictions: What You Can Compile

The 10,000-copy limit and the patent threat are only the first half of the story. Most programmers did not notice that Borland C++ 3.1's no-nonsense license statement contains language that restricts what kind of programs you can compile and distribute. You are restricted from developing:

...a compiler, development tool, environment product or library which includes any of the libraries, DLLs or source code included in this package...[or]...a product that is generally competitive with or a substitute for any Borland Language product.

How many of you 3.1 users knew that? You didn't read your no-nonsense license statement, did you? See what it says? You can't develop a programmer's editor because it would compete with Brief. You can't develop a compiler, an IDE, a profiler, a user-interface class library (such as D-Flat++), a resource compiler, and so on.

The patent stuff, which caught everyone's eye, drew attention to these other restrictions. Most of the programmers spoke out as if the noncompete conditions were new to version 4.0. They were not. Nonetheless, users were mad about the noncompete stuff too.

The Paradox Paradox

What was the intent? According to Smart, Borland did not want to restrict you in any of the ways that I just described,

Every programmer understands that "software patent" is an oxymoron

even though the language in the new no-nonsense license statement said otherwise. It merely wanted to prevent anyone from buying the Paradox engine, putting a user-interface shell around it, and selling a product that competes with Paradox. There's the real paradox, folks. The languages department wants to provide developers with a comprehensive database engine, but the applications department does not want those developers to use it in ways that the company does not approve of.

More Nonsense

Borland's shot in the foot was a double-barreled blast. The second barrel on their no-nonsense license statement contained the condition that the program you develop "...may not be an operating system."

Wow. I didn't know that Borland was planning to release an operating system. My earlier speculation would not apply to this one. The other large litigious competitor doesn't use Borland's compiler to compile their operating system. That can't be why Borland put this restriction in. My usually reliable sources weren't telling, either, other than to say that one of the lawyers added the language. Makes you wonder. Doesn't anybody outside of the legal department read this stuff before it goes in the big blue and white box?

This operating-system restriction had wide-ranging implications. For one thing, it ruled out UNIX ports. But worse, it hit embedded-system developers squarely between the eyes. An embedded system does not usually use MS-DOS, DR-DOS, or any other general-purpose operating system. The embedded program will be self-contained,

which means that it includes an operating system. You couldn't write one of them according to the new terms.

Of course, another outcry was heard 'round the world. Borland reacted quickly by saying in its CompuServe Q&A, "The restriction against creation of an OS is deemed unnecessary and will be dropped."

Why was the restriction necessary one day and not the next? One theory involves Borland's agreement with Microsoft. Borland has a license to distribute certain Windows development materials that are covered by Microsoft's copyright of the Windows API. Without those materials, Borland's users would need to purchase the SDK to develop Windows programs. The theory speculates that Microsoft granted that license on the condition that Borland would somehow prohibit its users from developing operating systems that compete with Microsoft. I do not believe this theory. Microsoft does not have similar restrictions on your use of its own software development products. I believe that Microsoft grants those licenses because its best interests are served when you develop Windows programs regardless of the compiler that you use. No, the story that Smart told me makes more sense. One lawyer, who doesn't know what an operating system is, put the language in, and no one else was smart enough to cross it out.

So, once again, why did the urgency of this operating-system condition disappear so fast? Because it isn't important, and you, their customers, howled, that's why.

The Healing of the Wound

Borland lost a large measure of credibility during this episode. It stuck a toe in to test the patent waters and got it shot off. It tried for some reason to limit the development of operating systems and got the door slammed shut in its face.

To regain some lost esteem, Borland went into damage-control crisis mode. The spin doctors rewrote the no-nonsense license statement to remove the operating system and 10,000-copy restrictions and to water down the non-compete clause to reflect their true, original, noble intentions, which are more palatable. The only restriction is as follows:

Your programs may not be merely a set or subset of any of the libraries, code, Redistributables or other files included in this package.

That restriction seems reasonable and reflects Borland's responsiveness to the concerns of its customers. More impor-

tantly, it demonstrates the power of the user's voice when a vendor tries to impose unreasonable restrictions on its customers. We, the programmers, won this one by the sheer force of our numbers. I hope that all vendors are watching and that they will have the good sense to let some users look at what the lawyers write before they commit to it.

We hope that Borland learned that lesson. In its zeal to counter their enemies, it forgot who its friends were. It showed us a different face, one that we had not seen before, a mean-spirited one that holds and can enforce software patents if it wants to. We want to believe that the old face has returned and that it is the true one.

Borland is not a litigious company. It has never sued anyone. Borland thought that reputation would hold it in good stead in the face of public reaction to their actions. But when I asked about the future, when the empty suits change occupants, when I asked about how we could be sure that some future regime would continue to overlook those fascist and burdensome no-nonsense license restrictions, Borland could not answer. In the face of overwhelming public disapproval of their actions and the hidden agenda that those actions seemed

to reveal, Borland did what it had to do. It took it all back. The version 4.0 no-nonsense license statement is, if anything, more liberal and more absent of nonsense than that of version 3.1.

What We Learned

This episode teaches us something else, too. Read the licensing conditions on whatever software-development tool you use to develop a program that you plan to distribute. Virtually all C++ compiler products have some restrictions. They require that you put a valid copyright notice on your software and do not remove any copyright notices that they include on the redistributable components. You indemnify the vendor from any liability if your programs do not work. You may redistribute the redistributables only as a part of an operating program and not as redistributables themselves. You must be a registered user of their product to distribute programs compiled with their product.

Only Borland and Microsoft have restrictions about what the programs themselves may do. Borland does not want you to distribute sets and subsets of its redistributables, whatever that means. Microsoft does not want you distributing programs that use its libraries,

MFC, and VBX redistributables in programs that programmers use to build programs that use VBXs. Interestingly, although Symantec C++ Professional licenses the MFC libraries from Microsoft for just such a purpose, it does not have a similar license restriction about what you can do with them. As you can see, it gets complicated.

Patent-Leather Agenda

Several years ago, every issue of every automobile magazine was sure to have at least one editorial where the author whined about the national 55-mph speed limit. Until the law was repealed those magazines acted as the self-appointed guardians of our right to drive fast. Similarly, every gun magazine today can be depended upon to wedge their editorial agenda against enemies of the people such as Janet Reno and James Brady who would abridge our Second Amendment right to own and bear semi-automatic assault weapons and handguns (in a well-formed militia, of course).


We in the programming-trade press are coming to sound very much like those other self-interest watchdog publications. We are beating this issue of software patents to death. What is the point?

Developer's Toolkit

ZyINDEX

Fastest, Most Powerful Text Retrieval Technology Available

Based on ZyINDEX—leader from the beginning in PC text retrieval



- Search 1 GB in Less Than 5 Seconds
- Up to 50 Million Documents, 10 GB Total Per Index
- Powerful Searches: Word, Phrase, Proximity, Boolean, Wild Cards and More
- Works Directly with MS Word, WordPerfect, AmiPro, dBASE, ASCII, and Others

Ideal for use with high-level application development environments such as Visual Basic, ToolBook, KnowledgePro, and ObjectVision.

Windows, DOS, and UNIX/386 libraries available \$3,995

Call for Specs and Demo.
800-544-6339

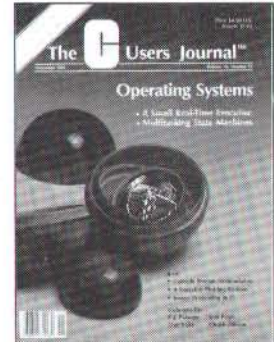
ZyLAB
Division of
Information Dimensions, Inc.

100 Lexington Drive • Buffalo Grove, IL 60089
Phone: (708) 459-8000 • Fax (708) 459-8054

CIRCLE NO. 329 ON READER SERVICE CARD

ADVANCED SOLUTIONS FOR C/C++ PROGRAMMERS

A FREE issue of **The C Users Journal** is yours—call now and ask for a trial subscription. If you like the code-intensive C/C++ programming solutions you find in your FREE issue, pay only \$29.95 for a full year's subscription. If not, write "cancel" on the accompanied invoice and owe nothing. There's NO RISK and NO OBLIGATION.



Try a **FREE ISSUE** The **C Users Journal**[™]

1601 West 23rd Street, Suite 200
Lawrence, KS 66046 USA

CALL: 913-841-1631
FAX: 913-841-2624

Overseas orders must prepay (\$65-U.S.) VISA, MC, or WIRE TRANSFERS

16-4B

CIRCLE NO. 908 ON READER SERVICE CARD

Well, in the first place, the issue is a technical one that is being administered with nontechnical criteria, and we, the trade press, are the only public forum that has or will tell the truth. The arguments for software patents are based in power, money, and politics. When you apply nontechnical, interim solutions to a technical problem, you almost always arrive at a final solution that does not work, if only because the technical parts of the problem are unsolved. Unfortunately, we are singing to the choir. You, our readers, already understand.

Every programmer understands that "software patent" is an oxymoron. The lawyers who prepare and file the patent documents do not understand. Neither do the Patent Office bureaucrats who grant the patents. Some of the people who hold the patents understand, but they are motivated by things other than technical purity, such as the promise of gain.

One of our smartest programmers is Bill Gates. His plans for Microsoft include 100 new software patents per year. He knows better, knows that the system does not know better, and plans to use that advantage to expand his power, influence, and wealth. Why am I surprised? Isn't this supposed to be the greed-is-good generation?

Software patents are everywhere. Most of the software-tool vendors who bring their demos to the *DDJ* conference room proudly announce that they have filed patents on parts of their products. I don't think they read our editorials. I recently attended a briefing of a new version of a well-known database management system. The vendor has a patent pending on his particular use of the B-tree algorithm in the indexes that support interfile relationships. I had to laugh, because years ago I used an identical technique in government software systems. It was obvious then. It is obvious now. The patent will probably be granted.

And yet, we keep thumping the drum. If we educate you about the dangers of software patents and their potential to compromise your livelihood, then we have done some good. At least you will be prepared. If enough of us kick up enough of a fuss, maybe our legislators will get the hint and do something positive about the problem. Maybe we can get the attention of those who need to understand their business and ours a little better. The effort might be in vain, however. Even if we educate the lawyers, they will pretend to continue to operate in a cyberfog. Technical ignorance supports their agenda, which

is collecting fees for knowing the law. Educating Patent Office bureaucrats is probably a waste of time, too. As soon as one of them understood software well enough to do the job, they would quit and find work as a programmer. Who wouldn't? And finally, trying to educate wannabe wealthy patent holders and fee collectors is guaranteed to be folly. They have already learned all that they need to know.

Goodbye, Sonny

I want to tell you about an unsung hero in our industry, someone who will never be the subject of a book, who will never receive a prestigious award, and about whom you will never hear, except today in this column.

Almost six years ago, in my first "C Programming" column, I told how my brother Fred got me started with C. He was a microcomputer pioneer with an engineering degree and a love of programming. He had every issue of *Dr. Dobb's* and was among the first of the homebrew computer makers. He was not one of the famous hackers, but he knew more about it than most. He kept a low profile and kept in touch with everything that was happening. You did not know about him, but he knew about all of you.



How to Become a More Effective C++ Programmer

In this age of ever more complex programming requirements, it's tough enough just to keep up with the times. To get ahead, you need to team up with experts in the field.

TurboPower Software has been writing award-winning programmer's tools since 1985. With all of them, you get complete source code, no royalties, plenty of examples, comprehensive help systems, literate documentation, easy learning curves, support for the latest compiler versions, and affordable prices. And you won't find better technical support anywhere.

"Honest-to-God power products and superior support."

Robert A. DelRossi, Director of Information Systems,
Liberty Real Estate Group

Call, fax, e-mail, or write today
for our informative catalog.
Evolution was never this easy!



Async Professional for C/C++ - comprehensive DOS serial communications, plus ZIP and LZH data compression. \$189



B-Tree Filer for C - fast, efficient, multi-user database toolbox, plus network functions. \$249



Object Professional for C++ - comprehensive user interface class library for DOS text mode apps. Includes screen painters for data entry and menu systems, plus much more. \$249



TSRs and More - state-of-the-art TSR engine for swapping and standard TSRs, swapping spawn, virtual arrays, and more. \$149

These toolboxes support all recent versions of Borland C++ and Microsoft C/C++.

TurboPower Software
"Your Satisfaction Guaranteed"



PO Box 49009 Colorado Springs, CO 80949-9009 USA
Toll-free: 800-333-4160 International: 719-260-9136
Fax: 719-260-7151 CompuServe: 76004, 2611

It was 1971. I was pounding out Cobol accounting programs when Fred dropped by. He brought a small aluminum hobby box with a front panel sporting four LEDs, four toggle switches, and some push buttons. It was a home-built computer, about the size of a cigar box, running an Intel 4004 microprocessor. I had never seen such a thing. We spent all afternoon loading programs and data into the small memory with switches and buttons and reading the output as binary values in the lights. The 4004 was meant to be used in calculators, but Fred was using it for some kind of black-box application in what we would call today an "embedded system." We got excited about that little box with four data lines and 256 bytes of memory. Someday, we thought, everyone would want one.

Fred grew with and ahead of the technology, always among the first to try new things. He built one of the first Altairs. It's still in his basement lab, still running. He recruited me to write programs for his projects and showed me how to squeeze code into tight spaces, citing Stevens's first law of programming, which said that any program can be reduced by one byte, and Stevens's second law, which said that sometimes Stevens's first law had to be applied recursively. Together we built many diverse embedded systems: a telephone call accounting system, a point-of-sale monitoring device, a power-company remote-station monitoring system, a laboratory etching device. We integrated microprocessors with PBXs, VCRs, TV cameras, cash registers, voice synthesizers, pagers, stepper motors, plating chambers, motion detectors. Fred designed and built the hardware, and I wrote the programs. We worked side by side, days and nights, and every project was a learning experience. Those old 8080 machines served as prototypes for the products and primitive development systems for the firmware. We typed the source code into memory with a Tele-Type terminal, programmed EPROMS from paper tape, and erased them under UV light. We wire-wrapped and soldered and patched and programmed and hand-assembled our way through dozens of one-of-a-kind machines, each one a wonder to behold and every one finished and performing its mission, some of them still in service today.

Twelve years ago, Fred's diabetes took him out of the action. With the passage of time he lost most of his vision, his kidneys, his legs, and a hand to the ravages of the disease. Not able to see well enough to design and debug hardware again, he returned to software, learning UNIX, Fortran, C, and assembly

language. His reading and typing were slowed sometimes to a crawl, but he never gave up, always maintained a hearty sense of humor, and never lost his enthusiasm for the work. Even when he could barely lift himself out of bed, he talked about ideas for the next project and held onto the belief that he'd lick the odds and see it through one more time. With his right hand gone and unable to see, he was still at it, figuring out how to integrate a joystick keyboard-emulator program with a voice synthesizer so that he could get back to programming as soon as he got well.

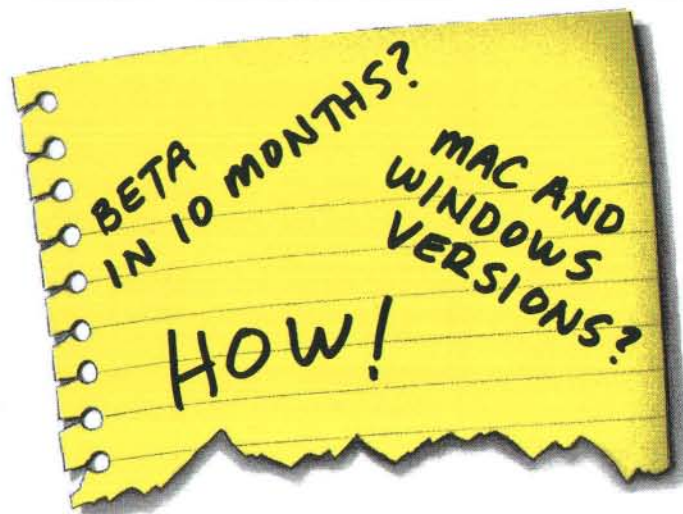
On his fifty-sixth birthday, eight days before Christmas, his frail body gave

way to a last heart attack, and Fred died, and his monumental spirit, intelligence, and courage were gone.

This is a lonely time for me. Everything that I know and all that I have done that is good can be traced in one way or another to things that my big brother Fred gave me. By his teaching, his example, and his encouragement, he was my mentor, my friend, and my biggest fan. But the loss is not mine alone. He left a family that he loved unconditionally and many loyal and devoted friends. We will all miss him.

DDJ

To vote for your favorite article, circle inquiry no. 12.



TURN TO TURNING POINT

For a very simple reason - We understand the business of software product development and we've been doing it successfully since 1983.

Our approach to project management insures proper controls, scheduling and budgeting. With a full time Quality Assurance and experienced Software Engineering staff, we have the technical excellence and proven track record to get your product done right. Plus, our experience in Macintosh®, MS-Windows® and DOS cross platform development is unparalleled.

Turning Point Software, the one to turn to for software product development at its best.

Call or write for a free portfolio.



One Gateway Center
Newton, Massachusetts 02158
617-332-0202

CIRCLE NO. 609 ON READER SERVICE CARD

Energize Your DOS Applications with a Windows-Style Interface

Are your DOS applications starting to show signs of age? If so, it's time to do some remodeling. With our new **C/Windows Toolchest™**, you can easily create interfaces for your DOS applications that are similar to Microsoft® Windows™ applications.

Two simple function calls are all it takes to create movable, resizable **windows**, complete with *scroll bars* and other standard **controls**; including *minimize*, *maximize*, *restore*, and *menu buttons*. A wide variety of other controls are also available; including *push buttons*, *radio buttons*, and *check boxes*.

Of course there is extensive support for **menus**. Create horizontal and vertical menus, with or without scroll bars. Arrange menu items in a single row or column, or in multiple rows and columns. Attach a *sub-menu* to any menu item.

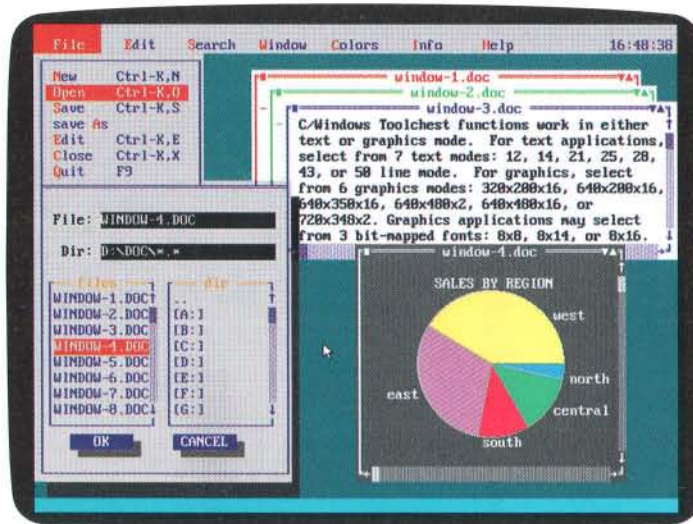
For collecting user input, you get a comprehensive set of functions to manage **data entry fields**. Collect data one field at a time, or all at once through complete data

entry forms. Use *picture clauses* to build data entry templates. Valid input may be enforced automatically, or you can define your own input validation functions.

Handling **mouse** input is easy. There are numerous high level mouse functions, including one that retrieves all mouse events and key-strokes. Mouse input is handled automatically by control buttons, menus, and data entry fields. Low level mouse functions are also provided just in case you need them.

In all, the

C/Windows Toolchest™ contains more than 250 functions to help you design a state-of-the-art user interface. Included are functions for implementing *context sensitive help*, *keyboard control*, and *graphics*. You also receive the complete source code for a multi-window **Notepad** editor that works in both text and graphics mode. C/Windows Toolchest™ works with C and C++ compilers from Mix®, Borland®, and Microsoft®.



Now One Interface Library Lets You Create Both Text and Graphics Mode Applications

☐ Please send FREE brochures

Disk Size: ☐ 5.25" ☐ 3.5"
(Requires DOS 2.0 or higher)

Name _____

Company _____

Street _____

City _____

State _____ Zip _____

Country _____

Telephone _____

☐ C/Windows Toolchest.....\$39.95

☐ Library Source.....\$10.00
(Source code for C/Windows library)

Shipping & Handling.....\$ _____

(\$5 USA, \$10 Canada, \$20 Foreign)

Texas Sales Tax (8.25%).....\$ _____

Total Amount of Order.....\$ _____

Paying By: ☐ Check or Money Order

☐ Visa ☐ MC ☐ Amex ☐ Discover

Card# _____

Exp. Date _____



**60 Day
Money-Back
Guarantee**

7 x 9, 624 pages

To Order Please Call:

1-800-333-0330

Orders or Technical Questions:

Tel: 1-214-783-6001

Fax: 1-214-783-1404



Mix Software
1132 Commerce Dr.
Richardson, TX
75081

Searching for a Search Engine

Tom Swan

Selecting the right tool for the job is always important, whether you are a carpenter, mechanic, or programmer. Too often, however, programmers choose algorithms for the wrong reasons—selecting a Quick sort because they believe it's always the fastest (not true) or using a binary search because they heard it always makes the fewest comparisons when finding elements in a sorted array (also not true). Never choose an algorithm because of its popularity. Depending on your application's requirements, a less-well-known method may be faster or more efficient.

On the other hand, it's human nature to be taken in by claims of superiority, as I discovered while searching for a tool of another variety—I'm talking oil-filter wrenches, now, not algorithms. You see, I need to regularly change the oil and filter in the diesel engine on board my home and sailboat, but it took three tries to find a wrench that would properly unscrew the filter can. The first tool I purchased, the most popular design, came with a band of steel attached to a vice grip that dented the filter case with only minimal pressure. The next sported a plastic strap and the written promise that "one size fits all." Imagine the raw holding power of plastic on a greasy canister, and it's not hard to understand why this filter wrench of the future could never work as advertised. (Products like these make me question whether tool manufacturers ever try their own wares. I often wonder the same about software vendors.) Finally, while poking around in a mechanic's tool chest, I found a homemade pipe, fitted for a socket wrench, with a rough leather strap that grabbed the filter the first time. Later, I bought one from the mechanic. This just goes to show that you should never choose tools based on their popularity or advertising claims. It's often the unlikely junk in the bottom of the drawer that works best.

Fast Failures

The same is true of algorithms. For instance, in dusting off an old program that I use to prepare Pascal listings for publication, I wondered whether a binary

search was the best way to look up entries in a sorted list of keywords—the critical code in this application that parses Pascal programs and converts keywords to lower case, optionally delimited for boldfacing in a word processor. I knew that a binary search makes only $\log N + 1$ comparisons, where N is the number of words in the array. Finding an entry in a list of 100 keywords, then, requires a maximum of six comparisons, which I wrongly assumed to be the best results I could expect.

To improve the program's speed, I considered using a hash function or a binary tree to search for keywords, but then I realized that, once again, I had been searching for a search engine for the wrong reasons. Most strings in a program listing are not keywords, so my program's speed was more dependent on how fast a word was not found than it was on the speed of a successful search. In other words, I needed a method that *failed* faster than the competition. Once I came to that realization, I found a way to boost my program's run-time speed by 20 percent. The algorithm that I chose, called a *trie search*—after "information reTRIEval"—is no faster on the average than a binary or hash function, but it requires a maximum of N comparisons—where N is the number of words beginning with the same letter—to determine that a word is not in the table. In practice, most failed searches take only one or two such comparisons—many take none—far better than required by a binary search, which tends to make the maximum number of comparisons for unrecognized words. By selecting the right tool for the job, tak-

ing into consideration the fact that most searches could be expected to fail, I increased my program's speed by using a less-popular, but better-suited, search algorithm.

Trie-Search Algorithm

A classic trie-search algorithm relies on a table arranged as illustrated in Figure 1. The figure shows only a portion of a complete table, indexed in the first column from A to Z. You could also index the table using other character sets—a standard ASCII trie table, for example, might have 127 rows. Each element in the index contains the number of another array that stores the table's words. The table entries might directly store data, or they could contain pointers—the exact format of the table depends on your program's requirements and the programming language you are using. A zero or null entry in a column indicates there are no words beginning with that letter. There are no Pascal keywords beginning with H, Y, or Z, so those entries are set to zero. (I'm using Borland Pascal's keywords here.)

As you can tell from Figure 1, a program can use a trie-search table to quickly determine whether a search argument is not a key word. In fact, no string comparisons at all are required for entries beginning with H, Y, or Z. Only one comparison is needed to find words beginning with B. To achieve the same results using a binary search requires up to six comparisons for negative searches of Borland Pascal's 57 keywords. In other applications with larger tables, you could extend the algorithm to use two or more tables indexed on a word's successive letters.

	[1]	[2]	[3]	[4]	[5]	[6]
[a]	2	and	begin	far	goto	xor
[b]	3	array	0	file	0	0
⋮	—	asm	—	for	—	—
[f]	4	0	—	function	—	—
[g]	5	—	—	0	—	—
[h]	0	—	—	—	—	—
⋮	—	—	—	—	—	—
[x]	6	—	—	—	—	—
[y]	0	—	—	—	—	—
[z]	0	—	—	—	—	—

Figure 1: Classic trie-search table.

OLE 2.0 Made Easier with MFC Version 2.5

Performance-tuning Windows NT™-based Applications

Harnessing the Windows™ Media Control Interface

Ask for **MSJ** at your
local newsstand

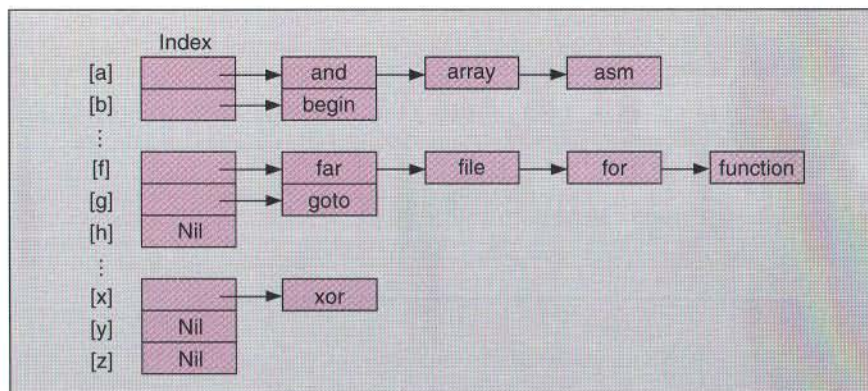


Figure 2: Trie table converted to a sparse matrix.

The trick is to minimize the number of full comparisons required to find words in the table or to determine their absence. Once you've structured the table, the rest is easy.

One problem, however, is evident from Figure 1. Many table slots are empty, wasting space. To minimize memory use, you can instead construct the table as a sparse matrix, as illustrated in Figure 2. Now the first column becomes an array of pointers, each of which addresses a list of words beginning with the same letter. (The table could be compressed somewhat by deleting the first letter of each word.) Entries with no words are null. As in the classic table, you could extend the sparse matrix by building other indexes for subsequent letters in each word. Carrying that idea to the extreme reduces the trie table to a digital list—that is, a binary tree of letters, with paths forming the table's words. Small tables such as the one shown here, however, work just as well with a single-level index.

Example 1 is pseudocode for Algorithm #18, Trie Search. The algorithm simply looks up an input argument's first letter in the index, then searches the linked list for a match. Only a single exact-match string comparison is needed inside the inner loop—the key ingredient of this method's speed. A binary search requires alphabetical less-than or

greater-than comparisons, further slowing searches for arguments not found.

Pascal Parser

Listings One, Two, and Three show the source code for my Pascal Parser, IDENT.PAS. SEARCH.PAS, the Pascal unit in Listing One (page 143), implements the trie-search algorithm. Keyword lists are composed of linked records of type *ResWordRec*. The global *Index* array corresponds to the *Index* column in Figure 2. Procedures *AddList* and *AddWord* build the trie-search tables—you can use these procedures to construct a trie-search engine for any list of words, but the words must be inserted in alphabetical order (see function *Initialize*). Function *IsReserved* determines whether a given word, passed as argument *Ident*, is a member of the table.

The other two listings, COMMON.PAS (Listing Two, page 143) and IDENT.PAS (Listing Three, page 143), use the trie-search engine to parse a Pascal listing. The program converts to lower case all keywords in a Pascal source file, and also optionally capitalizes all non-keywords (specify option -c). Use option -b to add <*> and <*> delimiters to keywords. The word *begin*, for example, is translated to <*>begin*>. Use the -b option only on a copy of a source file—after conversion, the file will no longer compile. (You can restore the original text by deleting all instances of <*> and <*>.) I use WINWORD.MAC (Listing Four, page 145) in Word for Windows to convert delimited words to boldface after inserting a listing into a document. You could probably whip up a similar macro for other word processors.

Your Turn

Next month, more algorithms. Meanwhile, send your favorite algorithms and tools to me in care of DDJ—software tools, that is.

DDJ

(Listings begin on page 143.)

To vote for your favorite article, circle inquiry no. 13.

```

input
  Arg: String;
var
  P: Pointer;
begin
  P ← Index[Arg[1]];
  while(P <> nil) do
  begin
    if P^.Word = Arg then
      return True;
    P ← P^.Next;
  end;
  return False;
end;
```

Example 1: Pseudocode for Algorithm #18 (trie search).

Tools.h++ Version 6.0

Now Internationalized!

**Collection Classes
Plus Much More!**

Tools.h++, the best selling industry standard C++ library, now includes Internationalization! A complete, efficient and versatile toolbox of over 100 C++ classes Tools.h++ will make virtually any programming job easier.

And now new Version 6.0 includes:

Internationalization

- Multi-byte and wide character strings
- Localized string collation
- Parse and format times, dates, and currency in multiple locales
- Support for multiple time zones and daylight savings rules
- Support for localized messages
- Localized I/O streams
- Many locales can be active simultaneously

Multi Thread safe

- Safe for use in multi thread environments
- Support for task specific data

Exception

- Comprehensive exception hierarchy
- Exception handlers can report in the local language

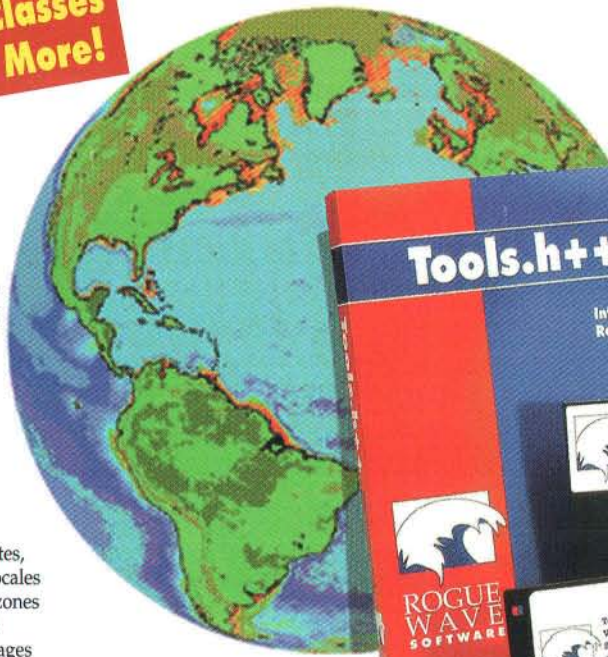
Comprehensive test suites are now available!

- ToolsPro.h++ includes a complete test suite for all classes

Now you can create one executable and ship it to multiple countries! Tools.h++ lets you read and print times, dates, numbers, and currency in the local format and language!

Includes template and non-template classes! You choose!

**Now also available on
WINDOWS NT and Macintosh**



Tools.h++ is a complete, efficient and versatile toolbox of over 100 C++ classes:

- String and Character manipulation classes.
- Singly and Doubly linked lists, Stacks, Queues and Vectors classes.
- Smalltalk™-like Collection classes: Set, Bag, SortedCollection, Ordered Collection, Dictionary, Stack Queue, etc.
- Regular Expression Class for search and replace.
- Tokenizer Class for easy string parsing.
- File Class to handle file manipulation with read, write, seek, erase, etc.
- B-tree Class to handle efficient keyed access of disk records.
- File Space Manager Class to allocate, deallocate and coalesce space within files.

- Virtual and Buffered Page Heap to manage objects bigger than 64k.
- All objects fully persistent.

Rogue Wave's Tools.h++ is an "industrial strength" library used in many commercial applications, small and large. All classes have not been derived from a single root object, so they can be easily integrated with other class libraries.

Tools.h++ is an excellent example of how to write true C++ code correctly. Source Code is included. All non-Windows classes are strictly portable between DOS and UNIX.

Other Rogue Wave Libraries Include:

View.h++

A complete C++ encapsulation of the industry standard OSF/Motif tool kit.

LAPACK.h++ and Math.h++

High level math libraries.

DB.h++

A portable C++ database interface for RDBMS's.

Our libraries work with most of the C++ compilers on the market today.



TO ORDER CALL 1-800-487-3217

P.O. Box 2328 • Corvallis, Oregon 97339 • 503-754-3010 • 800-487-3217 • FAX 503-757-6650

CIRCLE NO. 439 ON READER SERVICE CARD



RS-232⁴

Scores more touchdowns for your communications development in C and C++ for Windows, DOS, and OS/2

Comm++[™] 2.0

- ✓ C++ class library for Windows 3.1[™], MS-DOS[®], and OS/2.
- ✓ Device independence through inheritance. 47 big classes.
- ✓ Supports Microsoft C/C++, Visual C++, Borland C++ and other popular compilers.
- ✓ COM1.COM8, standard, and intelligent multiport boards.
- ✓ Baud rates to 115,200; drives 8250, 16450, and 16550 type UARTs including FIFO trigger level controls.
- ✓ XON/XOFF, RTS/CTS, and DSR/DTR handshaking.
- ✓ XMODEM, YMODEM, ZMODEM, Kermit, and ASCII file transfer protocols; 1K, G variants, RLE data compression, ZMODEM crash recovery, subdirectory recursion, more.
- ✓ VT52, VT100, ANSI & TTY terminal emulation classes.
- ✓ Modem control class includes handy `WaitForOK()`.
- ✓ Screen drivers include `TextWindows™`, `DataWindows`, and compiler independent "Vid" function set.

NEW! CommLib[™] 5.0

- ✓ Enhanced support for MS Windows includes language independent DLL and message notification for communication events (`WM_COMMNOTIFY`).
- ✓ Level 2 device independent functions for these drivers: **MS Win32**, MS Windows, Greenleaf Standard, Fast, Smart DigiBoard[™], Smart Arnet[™], Smart Star Gate[™], Sparkle[™], BIOS, Extended BIOS, Polled mode, **MODEM Assist Plus™**, and FOSSIL.
- ✓ 286 and 386 Protected Mode for Phar Lap and Rational Systems DOS extenders.
- ✓ **CompuServe B+**, XMODEM, YMODEM, ZMODEM (with crash recovery), Kermit, ASCII file transfer protocols.
- ✓ Unlimited number of ports for ISA, EISA, and MCA.
- ✓ Modem controls, keyboard, screen, RTS/CTS, DSR/DTR & variable level XON/XOFF handshaking, and much more.
- ✓ *Exclusive:* Mix different multiport boards in a PC!

ViewComm[™] 3.0

- ✓ Full featured RS-232 "Datascope" with graphic breakout box for PC or laptop. Uses 1 or 2 serial ports.
- ✓ Monitor, capture data in ASCII, EBCDIC, Baudot to 115,200 baud. Supports 8250, 16450, and 16550 UARTs.
- ✓ Capture to buffer or file, review, string search.
- ✓ Unlimited triggers on modem status, data patterns, including wildcards!
- ✓ View data in hex, octal, decimal, binary, with graphic control codes. Special fonts provided for EGA and VGA.
- ✓ Variable resolution timestamps for each character, down to 1 msec. optionally stored with data.
- ✓ Source mode lets you send from file, keyboard or both.
- ✓ Users say it easily pays for itself the first time you use it!
- ✓ Cables, setup guide, and reference manual included.

PowerComm[™] 1.1

- ✓ **PowerComm Toolkit 1.1** for Windows 3.1 provides CommLib DLL and VxD. (**Requires CommLib 5.0**)
- ✓ Drives most standard multiport boards; share board among multiple Windows 3.1 and/or DOS apps.
- ✓ Throughput multiplier—much faster than Windows or other driver. The VxD is written using CommLib's fast assembler code in 32-bit flat model.



- FREE Source code—all of it!
- FREE Unlimited Tech Support.
- FREE online help system.
- FREE access to Greenleaf BBS.
- Professional quality, same day shipment, personalized service.
- Greenleaf Gold Support also available—ask about it!
- 60-day money-back guaranty (if source code not opened).
- Other C and C++ programming tools available.



Call Greenleaf Software today for complete information or to order. Major credit cards, COD, approved purchase orders. Same day shipment in most cases.

(800)523-9830



(214)248-2561 FAX: (214)248-7830
BBS: (214)250-3778 for free demos and information.
16479 Dallas Parkway, Suite 570, Dallas, TX 75248

Greenleaf CommLib 5.0 NEW!
Greenleaf CommLib Professional
Greenleaf PowerComm 1.1
Greenleaf Comm++ 2.0
Greenleaf ViewComm 3.0

\$359
\$538
\$179
\$249
\$399



Think Globally, Act Locally: Inside the Windows Instance Data Manager

Klaus Müller

Introduction

by Andrew Schulman

Many DOS programmers still "don't do Windows," seeing it as irrelevant to DOS programming. But this is unrealistic, because Windows Enhanced mode affects even software loaded *before* Windows is loaded, including DOS memory-resident programs (TSRs), device drivers, and even DOS itself. In short, many DOS programs have no choice but to become Windows-aware.

Windows awareness is especially important for "instance data." For example, load a TSR like Chris Dunford's CED command-line editor and then start Windows Enhanced mode. Open two DOS boxes. Type a command in one DOS box, switch to the other one, and then press the up-arrow key. The command typed in the first DOS box appears in the second, as "state" leaks across! This unintentional interprocess communication might appear to some programmers as a feature, but it is more likely to strike users as a bug. A DOS programmer who blows off this problem with a proud "I don't do Windows" had better be sure that every one of his users feels the same way.

Now put the statement LOCALTSRS=CED in the [Non-WindowsApp] section of SYSTEM.INI (if it's not already there), and restart Windows. This time, commands typed in one DOS box are recalled only in that DOS box; they don't leak across into the other one. As its name implies, LOCALTSRS= has somehow made CED's state "local" to each DOS box. Exactly how this works is the subject of this month's "Undocumented Corner."

Rather than use CED, you can switch to the DOSKEY utility that Microsoft includes in DOS 5 and 6. This command-line editor exhibits the same correct behavior as LOCALTSRS=CED, except that no LOCALTSRS=DOSKEY statement is necessary. Clearly, DOSKEY is doing something CED isn't.

Unlike CED, DOSKEY intercepts INT 2Fh and looks for calls to AX=1605h and AX=4B05h. If it receives a call to either of these functions, DOSKEY declares the address and size of its command-line history buffer as "instance data"—that is, as data that must be local (rather than shared) in each DOS box. Note that "instance data" in this context has nothing to do with multiple "instances" of Windows applications (though there are some analogies).

INT 2Fh AX=1605h is documented in the Windows Device Driver Kit (DDK). This is a crucial interface with which DOS programmers must be familiar. It may seem perverse that an API necessary for DOS programmers is located in the Windows DDK, but INT 2Fh AX=4B05h, documented as "Identify Instance Data" in the *MS-DOS Programmer's Reference*, is identical (at least as it relates to instance data). In fact, DOSKEY uses the same piece of code to handle both calls. Unfortunately, the *MS-DOS Programmer's Reference* indicates that INT 2Fh AX=4B05h is related to the relatively unused DOS task switcher and says nothing about the need for DOS programs to instance data for compatibility with the far more prevalent Windows Enhanced mode.

But what does INT 2Fh AX=1605h actually do? And how does it relate to other means of instancing data, such as the LOCALTSRS= statement (or its LOCAL= equivalent for

There are programmers who see Windows not as a graphical user interface, but as an operating system with preemptive multitasking of virtual machines (VMs). These developers have to worry about things like hardware interrupt handlers that operate in the context of a specific VM.

Asynchronous access of data in a specific VM is possible via the documented *CB_High_Linear* field in the largely undocumented VM control block (VMCB)

structure (see *DDJ*, January/February 1994). The current VM is mapped in the first megabyte of the linear-address range. You can access any VM's address space (current or not) by adding *CB_High_Linear* to its linear address.

So far, so good. But sometimes when you want to access data in a VM, you get a page fault. This page fault is transparent (applications don't see it), but it can lower performance and lead to serious problems inside an interrupt handler.

What's wrong here? It turns out that the page faults are an integral part of instance-data management in Windows Enhanced mode.

Local vs. Global Data

Instanced data is born as global data before Windows starts, but once Windows starts, it becomes local for each VM.

In general, it would be good if *all* data in a VM were local. In a genuine protected multitasking environment, global data is very dangerous. Consider an ugly TSR that crashes a DOS system. In a truly protected environment, the multitasking kernel nukes only the crashed VM.

Windows 3.x Enhanced mode does not support this level of safety. Since Windows is based on DOS, Microsoft compromised. All memory allocated before Windows starts (including DOS TSRs, device drivers, and DOS itself) will be mirrored in each VM via the 386 paging mechanism. Thus, this memory is *global*, shared by all VMs.

There are several reasons to allow the presence of global data. For one thing, Windows rests on top of DOS, and some DOS data *must* be global.

Klaus studies information technology at the Dresden University of Technology's Fraunhofer Institut for Microelectronic Circuits IMS-2. He is currently developing a heterogeneous multiprocessor system for parallel image processing based on the TMS320C40 processor. Contact Klaus on CompuServe at 100117,2526.

(continued from page 125)

DOS device drivers)? Eventually these methods of declaring instance data to Windows, plus several others, lead to the `_AddInstanceItem` function provided by the Windows Virtual Machine Manager (VMM). This call is documented in the Windows DDK and in the book *Writing Windows Virtual Device Drivers*, by David Thielen and Bryan Woodruff (Addison-Wesley, 1994).

Okay, so what does `_AddInstanceItem` do? What is instance data really, and how does VMM implement it? The Microsoft KnowledgeBase includes a surprisingly good explanation, "Instantanced Data Management in Enhanced Mode Windows" (Q90796). However, this states that the internal "instance buffers are not accessible to VxDs or TSRs; they are local data structures to be accessed by the VMM only."

In this month's "Undocumented Corner," Klaus Müller shows how to access the internal instance-data structures, using a virtual device driver (VxD) loaded early in the Windows boot process, right after VMM. By using the documented `Hook_Device_Service` call to intercept the `_AddInstanceItem` function, Klaus's VxD builds up a picture of the instance-description buffer.

To further describe the Windows instance-data manager, Klaus uses another interesting method: examining the error-message strings that appear in the widely available debug version of WIN386.EXE. Many of these error messages refer to internal VMM functions whose names we otherwise would not know; see Figure 1.

Once the internal instance-data structures are located, the results must be *interpreted*. Let's say that (as in Figure 2) the virtual keyboard device (VKD) instances 28h bytes at address 415h. So what? Well, these 28h bytes include the BIOS keyboard buffer. VKD instances this buffer

so that each DOS box—actually, each virtual machine (VM), including the System VM in which Windows applications run—has its own local BIOS keyboard buffer. Keys typed in a DOS box don't leak into the user's copy of Excel or Word for Windows. This has a downside, too: It's difficult (though not impossible) for Windows applications and full-screen DOS boxes to deliberately "push" keystrokes into each other.

In previous "Undocumented Corner" columns (January and February 1994), Kelly Zytaruk examined the Windows virtual machine control block (VMCB) and noted that offsets 0BCh and 0CCh in the VMCB refer to instance data. Klaus fleshes out this point.

I expected that all of Klaus's results would have to be thrown out for Microsoft's forthcoming Chicago operating system (Windows 4). However, Klaus reports that the instancing mechanism has not fundamentally changed and that his programs for locating the instance-data structures work in Chicago, too. Of course, Chicago is still in prerelease, and anything can happen between now and when it ships. Klaus does note that VMM in Chicago provides a `_GetInstanceInfo` call, which reports whether a given region is instanced or not, though whether this is more useful than the existing `_TestGlobalV86Mem` is unclear. Future articles from Klaus will show how to locate device CB areas and asynchronously access instance data without causing a page fault.

In addition to the usual places to download DDJ code (see page 3), you can get programs and source code mentioned in this article from the new Undocumented Corner area in the DDJ Forum on CompuServe (GO DDJ). If you have any comments or suggestions for future articles, please post messages to me there, as well; my CompuServe ID is 76320,302.

(continued from page 125)

Consider the example of the DOS system-file tables (SFTs) when SHARE.EXE is loaded. The SFTs present before Windows started need to be visible to all VMs.

Global data also saves memory. Remember the days before 386 memory managers? A well-equipped system with network drivers, mouse drivers, disk-caching programs, and other resident software could consume 400 Kbytes of conventional memory. If you created three VMs, you quickly wasted 1 Mbyte of memory.

Wasted? What about the paging mechanism of 386-mode Windows with its ability to extend physical memory with disk memory? Unfortunately, TSR memory must often reside permanently in *physical* memory. Many TSRs maintain a hardware interrupt; paging out memory which contains hardware interrupt handlers would cause unpredictable results. The interrupt-handler code would be executed for each VM separately, so the best result from paging global TSRs would be a remarkable performance decrease. Consequently, memory belonging to DOS TSRs, device

drivers, and DOS itself is, by default, global to all VMs, and is not pageable. If a TSR changes some data in its memory area, the change takes effect in all VMs. If the TSR crashes a VM because of a memory-related error, all VMs will be crashed, including the System VM with all the Windows apps.

Although the executable code of the TSRs can be the same for all VMs, the data must be private for each VM. Such privacy is called "instancing." While this should not be confused with instance data in Windows applications, it is analogous.

Any portion of software loaded before Windows can be instanced using one of several documented techniques, including INT 2Fh AX=1605h and the LOCALTSRs= and LOCAL= statements in SYSTEM.INI. Some obvious candidates for instancing are the interrupt-vector table and parts of the BIOS data area. The keyboard buffer has to be instanced, as do the history buffers belonging to any command-line editors loaded before Windows. (Why doesn't the history buffer for a command-line editor loaded *inside* a Windows DOS box have to be in-

stanced? Answer: Because the memory is *already* local.)

Instance Data and Paging

The memory management of Windows Enhanced mode is based on the 80386 paging mechanism. The smallest unit of memory is one 4K page. The following types of memory are to be found in the VM's address range (the page types are documented in the DDK):

- Global data. Nonpageable, system-wide data shared by all VMs. Changing the data in one VM changes the data in all VMs. By default, the allocated DOS memory at Windows startup is PG_SYS. The memory is "mapped" into each VM. Page type: PG_SYS.
- Local data. Pageable, local data specific for each VM. The free DOS memory at Windows startup is local. A DOS program started in a DOS box cannot crash other VMs because of an error that belongs to its PG_VM memory. Page type: PG_VM.
- Instance data. A mixture of instanced and global data. Like PG_SYS pages, they are nonpageable. The differ-

ence from global (PG_SYS) memory is that some of the data is marked as local and handled in a specific manner. Page type: PG_INSTANCE.

Though PG_INSTANCE pages are not paged out to disk, PG_INSTANCE can be marked "not-present"; this is key to the instance-data mechanism. Only one VM at one time has a PG_INSTANCE page "present." The corresponding pages in the other VMs are marked not-present. If the pages were all swapped out during a task switch, the global data would become local; the pages would not be updated when another VM changed the global data. Conversely, if the pages were still present in the other VMs, writing data to the instanced part of the pages would make them global because all corresponding PG_INSTANCE pages have the same physical base.

Windows saves the instanced parts of PG_INSTANCE pages in a special buffer. Because the paging mechanism has 4K granularity, a physical copy is required for any instance data item smaller than 4K. An instanced data item can be as small as a single byte. Many individual instance items can thus decrease performance.

The Instance-Data Manager

The instance-data manager (IDM) manages the instance mechanism. It is part of the Windows Virtual Machine Manager (VMM) and exports the documented *_AddInstanceltem* service. While processing *_InstanceInitComplete*, the IDM allocates memory via the VMM *_PageAllocate* service for the following buffers:

- For each instance item, the instance-description buffer contains its linear address, its length, and the location of its data within the instance buffers. VxDs declare instance data via the documented *_AddInstanceltem* service and *InstDataStruc* structure. The code in VMM for *_AddInstanceltem* builds a linked list of these structures. At *_InstanceInitComplete*, VMM discards any duplicate instance-data requests (for example, if one VxD instances 2 bytes at 415h, and another instances 20h bytes at 400h) and builds the instance-description buffer as a sorted array of *InstMapStrucs*. During *Instance_Init_Complete*, the sorted list is examined. With the information from the list of *InstMapStrucs*, the IDM builds the instance-description buffer as an array of *InstanceMapStrucs*.
- The instance-snap buffer is used to save the instant data present at Windows startup time. When Windows exits back to DOS, the instance data is restored.

```

_InstanceInitComplete no instance list
Tells us there must be an instance list. This is obviously a chain of linked InstDataStrucs from
documented in VMM.INC.

_InstanceInitComplete entries not sorted #esi > #edi
_AddInstanceltem sorts the entries and chains them together via the InstLinkF and InstLinkB
fields of the InstDataStruc.

Computed Inst_VM_Buf_Size of 0 _InstanceInitComplete
_InstanceInitComplete must compute a Inst_VM_Buf_Size.

Allocation failure VM1 Inst _InstanceInitComplete
Allocation failure Inst snap _InstanceInitComplete
Allocation failure Inst Descrip _InstanceInitComplete
_InstanceInitComplete allocates space for the VM1 Inst buffer, the Inst snap and Inst descrip
buffer.

Fail grow Instance desc buff AllocateInstanceMapStruc
IDM function AllocateInstanceMapStruc tests the size of the instance description buffer and, if
needed, grows the size.

Swap_Instance_Page, 0 in Inst_Page_Owner for page #ecx
Inst_Page_Owner is the VM in which the PG_INSTANCE page is marked present.

Swap_Instance_PageFS ERROR INSTANCE PAGE > 10Fh
ERROR: Re-entered instance copy procedure
The IDM function Swap_Instance_PageFS copies the instance data in the instance buffer of the
VM and changes the owner of the PG_INSTANCE page.

ERROR: Instance fault on suspended VM #EBX
A suspended VM cannot own a instanced page.

Instance fault on page with 0 IMT_Inst_Map_Size
The Inst_Page_Owner will change after a page fault on a instance page is detected. The IDM
determines the new owner VM and saves the instanced data of the old owner via
Swap_Instance_Page.

_AddInstanceltem failed InstDataStruc @#EDI Create_Int2F_Inst_Table
The internal routine that processes the InstDataStruc chain from INT 2Fh AX=1605h is called
Create_Int2F_Inst_Table.

```

Figure 1: Some instance-related error messages from the debug WIN386.EXE.

```

C:\DDJ\INST>instwalk
V86 API from ListInst.386: Result of _AddInstanceltem Hook.
Summary of allocation calls to the Instance Data Manager.

```

Name of VxD	InstLinAddr	InstSize
VKD : Sys_Critical_Init_Proc	0x00000415	0x00000028
VKD : Sys_Critical_Init_Proc	0x00000471	0x00000001
VKD : Sys_Critical_Init_Proc	0x00000480	0x00000004
VKD : Sys_Critical_Init_Proc	0x00000496	0x0000000B
VMM: _Allocate_Global_V86_Data_Area	0x0002807C	0x00000374
V86MMGR : Unknown_Service	0x00028435	0x00000006
VTD : Device_Init_Proc	0x000284C0	0x00000008
VDD : Device_Init_Proc	0x00000449	0x0000001E
VDD : Device_Init_Proc	0x00000484	0x00000007
VDD : Device_Init_Proc	0x000004A8	0x00000004
VDD : Device_Init_Proc	0x00000410	0x00000002
VCD : Device_Init_Proc	0x00000400	0x00000008
VCD : Device_Init_Proc	0x0000047C	0x00000004
DOSMGR : Device_Init_Proc	0x00000413	0x00000002
DOSMGR : IGROUP	0x00000504	0x00000001
DOSMGR : IGROUP	0x00000000	0x00000040
DOSMGR : IGROUP	0x00001550	0x0000001A
DOSMGR : IGROUP	0x0000156A	0x00000072
VMM: _Allocate_Global_V86_Data_Area	0x000286B4	0x00000256
DOSMGR : IGROUP	0x00003CE0	0x00000004
DOSMGR : IGROUP	0x00001342	0x00000002
DOSMGR : IGROUP	0x00004830	0x000008F0
DOSMGR : Device_Init_Proc	0x00027FD0	0x00000010
DOSMGR: Instance_Device	0x00001278	0x00000004
DOSMGR: Instance_Device	0x0001EEC2	0x00000004
DOSMGR: Instance_Device	0x000283F0	0x00000004
VMM:Create_Int_2F_Inst_Tbl:DOSKEY	0x00017D30	0x00000288
VMM:Create_Int_2F_Inst_Tbl:DOSKEY	0x00018C53	0x00000200
VMM:Create_Int_2F_Inst_Tbl:MOUSE	0x00012158	0x00000889
VMM:Create_Int_2F_Inst_Tbl:MOUSE	0x00012AD7	0x0000010A
VMM:Create_Int_2F_Inst_Tbl:VLM	0x0000DD60	0x0000001C
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x00000500	0x00000002
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x0000050E	0x00000014
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x0000070C	0x00000001
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x00005140	0x00000002
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x000053B0	0x0000004C
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x00001252	0x00000002
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x00001262	0x00000004
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x00001429	0x00000106
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x00001530	0x00000001
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x000021F0	0x00000022
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x000012B9	0x00000001
VMM:Create_Int_2F_Inst_Tbl:SYS DRV	0x000012BC	0x00000002

Figure 2: Sample output from INSTWALK.

- The VM1 instance buffer initially contains a copy of the data in the instance-snap buffer.

When a new VM is created, it also gets a VM instance buffer, which contains all instanced data for the VM when the PG_INSTANCE pages are set not-present. The offset and handle of the VM instance buffer can be found at offsets 0BCh and 0C4h in the VMCB. In Chicago, the VM instance buffers (including the VM1 buffer) are part of the VMCB and are allocated via the *_Allocate_Device_CB_Area* service. (In a future article, I'll describe a VxD that hooks this call to help enumerate these CB areas.)

_InstanceInitComplete is an internal VMM function. While working with the debug version of WIN386.EXE, I found some very informative error messages that include references to this internal function; see Figure 1. A great deal can be learned about the internals of the IDM simply by examining these error messages and pondering what the IDM must require for normal, error-free operation. To verify the interpretation of the error messages in Figure 1, it is useful to inspect the code of the IDM. Since the error messages are issued via *Out_Debug_String* with the string offset in ESI, it is easy to locate the desired code that would issue this message if something went wrong.

From there, it is easy to work backwards to the code's normal operation.

To illustrate the handling of the PG_INSTANCE pages, assume that they are currently owned by the System VM (SYSVM, or VM1). When a second VM is created, the VMM begins to schedule the VMs according to their priority. If the VMM schedules from SYSVM to VM2, the PG_INSTANCE pages in VM1 are still present, and the corresponding pages in VM2 are not. If any code in VM2 accesses the

There are several reasons to allow the presence of global data

PG_INSTANCE pages (via the linear address), a page fault occurs. The IDM copies the SYSVM's instanced data to SYSVM's instance buffer and sets the faulting PG_INSTANCE page of the SYSVM as not present. Then the IDM sets the corresponding PG_INSTANCE of VM 2 present and copies the instanced data from the instance buffer of VM2 to the page.

This instancing mechanism is not complicated. But Microsoft has added an important twist for performance reasons: The physical copy of the instance data to the instance buffers occurs only if the pages are written (but not read) by a VM other than the one which owns the PG_INSTANCE pages. Because they are set not-present for the VM, a page fault occurs and the copy starts. Now we can understand why Microsoft has stated that "fragmented and large instance areas decrease the performance of the swapping mechanism."

In summary, for *write* access VMM will: 1. Copy instance data from the PG_INSTANCE page physical base of the former owner to the VM's instance buffer; and 2. copy instance data from the instance buffer of the owner to the PG_INSTANCE page physical base. For *read* access VMM will copy instance data from the instance buffer of the new owner to the PG_INSTANCE page physical base.

Spying on Instance Data

For a deeper understanding of the instancing, it is useful to write a program that displays the address and size of all instance data and the name of the program which asked for the data to be instanced. LISTINST.386 (see Listing One, page 146) is a VxD I wrote that initializes shortly after the VMM, before other VxDs gain control, and hooks the VMM *_AddInstanceltem* service at *Sys_Criti-*

COMPUTER
LANGUAGE
PRODUCTIVITY
AWARD
1992

C-Vision

Four Great C Tools in One Coordinated Package

- C-Xref for cross referencing
- C-Tree for function call analysis
- C-Lines for outlined listings
- C-Format to reformat C programs

Cross-Reference

Usage symbols:

- a symbol, function, or macro is declared
? - a function is declared by being used
& - symbol's address is taken: &abc
= - symbol takes on a new value: i=1; or i++;
u - macro is undefined: #undef COUNT

cos(double), returns double
trig2.c 3# 10

PI, macro: 3.14159265359
trig.h 1#
trig1.c 16 17
trig2.c 5

sin(double), returns double
trig.h
trig1.c
trig2.c

sin_1(double), returns
trig1.c

tan(double), returns
trig2.c

Eight Reasons to Choose C-Vision

1. Built with 386 DOS extender technology for high speed and so you won't run out of space.
2. Intelligent cross-referencing - can distinguish between declarations, uses and modifications of variables
3. Intelligent labeling of trees and cross-references provides the types and prototypes of variables and functions
4. Cross-reference information can be dumped to disk as ASCII files and manipulated directly
5. Uses the same syntax engine as the thoroughly tested PC-lint
6. Project file compatible with PC-lint
7. Numerous options for customization of output
8. "Stunningly bulletproof"

Computer Language, Sept.-1992

Hierarchy Tree

```
1: tan(double), returns double, trig2.c @ 8
2:   ↳ cos(double), returns double, trig2.c @ 3
3:   ↳ sin(double), returns double, trig1.c @ 12 [self]
4:     ↳ sin_1(double), returns double, trig1.c @ 3 [self]
       ↳ PI, macro: 3.14159265359, trig.h @ 1
```

trig1.c

```
1 #include "trig.h"
2
3 static double sin_1( double x )
4 {
5     double y;
6     if( 27.0 - 4 * x * x == 27.0 )
7     { return x; }
8     y = sin_1( x/3.0 );
9     return y * (3 - 4 * y * y);
10 }
11
12 double sin( double x )
13 {
14     if( x < 0 )
15     { return -sin(-x); }
```

Gimpel Software

3202 Hogarth Lane, Collegeville, PA 19426

CALL TODAY (215) 584-4261
Or FAX (215) 584-4266

Only \$139

Specify MS-DOS or OS/2

30 Day Money-back Guarantee.

PA add 6% sales tax.

C-Vision is a trademark of John Rex and Gimpel Software.

PC-lint is a trademark of Gimpel Software.

cal_Init time. INSTWALK is a DOS program that displays the information saved by LISTINST.386.

Two other required VxDs are VXD-QUERY.386 and (to examine instance data in Chicago) LISTCALL.386. All three VxDs are loaded automatically if you run the DOS program VXDLOAD.EXE just before starting Windows. (VXDLOAD uses the versatile INT 2Fh AX=1605h interface to tell Windows to load the VxDs.) While there isn't room to show all of the source code, the programs are available electronically (see "Availability," page 3).

For each call to *_AddInstanceltem*, LISTINST stores the address of the caller and the offset of the passed-in *InstDataStruc*. (This structure is documented in VMM.INC and INT2FAPL.INC, included with the DDK and with VxD-Lite.) LISTINST has a V86 API which allows DOS programs running in a VM to get the results of the *_AddInstanceltem* hook. INSTWALK uses this V86 API and prints out all instanced data.

Sample output is shown in Figure 2. To clarify what this output means, Figure 3 shows a hex dump (using the PROTDUMP utility discussed in the "Undocumented Corner," January and February 1994) of one of the instance data items declared by DOSKEY; clearly, this instance data is the DOSKEY-command history buffer. By specifying a VM number on the command line, PROTDUMP can view this buffer in each VM; see #1 and #2 in Figure 3. The ownership and purpose of the buffer is identical for all VMs, but the data (here, the command history) differs in each VM. This is precisely what instance data means.

The INSTWALK utility has a -p switch to dump out the instance page-ownership array (an internal IDM structure which lists all PG_INSTANCE pages and their current owners) and the instance-description buffer in raw form. Because this structure is not accessible, I built my own array. Remember, only one VM at a time can own a PG_INSTANCE page. So you have only to walk down the VM's page tables to determine the VM in which the PG_INSTANCE page is set present; see LISTINST.386 for details. This is similar to output from the .mi command available in debuggers such as WDEB386 and Soft-ICE/Windows when the debug WIN386.EXE is installed. INSTWALK -p also dumps out the offsets in the instance buffers, which is important if you want to access the data in the instance buffers.

LISTINST.386 gets the names of the callers to *_AddInstanceltem* via a service provided by VXDQUERY.386, which provides a complete map of the VMM and VxD address space. VXDQUERY

```
C:\DDJ\INST>instwalk | grep DOSKEY
VMM:Create_Int_2F_Inst_Tbl:DOSKEY 0x00017D30 0x00000288
VMM:Create_Int_2F_Inst_Tbl:DOSKEY 0x00018C53 0x00000200

C:\DDJ\INST>\ddj\protdump\protdump #2 18c53 200
00018C53 45 00 63 3A 5C 65 70 73 5C 65 70 73 69 6C 6F 6E E.c:\eps\epsilon
00018C63 2E 65 78 65 20 24 2A 00 69 6E 73 74 77 61 6C 6B .exe $*.instwalk
00018C73 20 3E 20 69 6E 73 74 77 61 6C 6B 2E 6C 6F 67 00 > instwalk.log
00018C83 67 72 65 70 20 44 4F 53 4B 45 59 20 69 6E 73 74 grep DOSKEY inst
00018C93 77 61 6C 6B 2E 6C 6F 67 00 5C 64 64 6A 5C 70 72 walk.log.\ddj\pr
00018CA3 6F 74 64 75 6D 70 5C 70 72 6F 74 64 75 6D 70 20 ot dump\protdump
00018CB3 31 38 63 35 33 20 32 30 30 00 5C 64 6A 6A 5C 70 18c53 200.\ddj\p
... etc. ...

C:\DDJ\INST>\ddj\protdump\protdump #1 18c53 200
83018C53 45 00 63 3A 5C 65 70 73 5C 65 70 73 69 6C 6F 6E E.c:\eps\epsilon
83018C63 2E 65 78 65 20 24 2A 00 69 6E 73 74 77 61 6C 6B .exe $*.cd tapci
83018C73 73 00 74 61 70 63 69 73 00 63 64 5C 70 72 6F 63 s.tapcis.cd\proc
83018C83 6F 6D 6D 00 70 63 70 6C 75 73 00 63 64 5C 74 61 omm.pcuplus.cd\ta
83018C93 70 63 69 73 00 74 61 70 63 69 73 00 67 72 65 70 pcis.tapcis.grep
83018CA3 20 4E 46 5F 20 5C 62 6F 72 6C 61 6E 64 63 5C 69 NF_ \borlandc\i
83018CB3 6E 63 6C 75 64 65 5C 74 6F 6F 6C 68 65 6C 70 2E nclude\toolhelp.
83018CC3 68 00 65 78 69 74 00 63 64 5C 69 6E 73 74 00 63 h.exit.cd\inst.c
... etc. ...
```

Figure 3: Examining DOSKEY's instance data.

For ease and convenience, these and other VNR books may now be ordered toll-free by calling 1-800-544-0550 (Dept. Z 1586) Or GO VNR on the CompuServe® Network.



VNR KNOWS SOFTWARE ENGINEERING

Why do breakthroughs on one software project not yield similar results on another? Why does software obey its own undisclosed laws? Finally these questions are answered in a book which gives you the tools to anticipate, avoid and correct defects when engineering software. **SYSTEMS, SOFTWARE AND QUALITY ENGINEERING** by Arthur E. Ferdinand (\$59.95 0-442-01730-8)

FOR A LIST OF THE BEST NEW BOOKS IN SOFTWARE TESTING

Whenever you develop software, it is essential that it is documented and reproducible at every stage of its development. **CONFIGURATION MANAGEMENT FOR SOFTWARE** by Stephen B. Compton and Guy Connor (\$39.95 0-442-01746-4) offers clear and simple procedures for documenting everything you do, every step of the way. It may not sound important until you experience the alternative.



Van Nostrand Reinhold
115 Fifth Avenue New York, NY 10003

Call VNR at
1-800-544-0550

TE Developer's Kit

Incorporate text editing features into your application easily and cost effectively. Features: multiple files/windows, word-wrap, edits large files, undo, cut/paste, printing, and search/replace. The word processing features include **bold**, underline, and *italic* styles. Paragraph features: **indentation**, **double spacing**, **centering**, and **justification**. The Windows and OS2 versions also offers **Wysiwyg**, **multiple fonts and point sizes**, **imbedded pictures**, and many more character styles.

Additional features for the Windows version: ruler, tab stops, pagination, RTF support, DLL and custom control interface, hanging indents, and more. Includes the complete 'C' source code. (DOS: \$299, Windows: \$349, OS2-PM: \$379)

ReportEase

Report Write/Mail Merge Engine

ReportEase consists of a report layout editor and a report executer. Advanced features include: multiple files, multiple sorts; data, calculation, system, and dialog fields; bold, underline, italic formats; subtotals, record filter, functions, word wrapping and more. Simple interface works with any application. Includes the 'C' source code. (DOS: \$349, Windows: \$379)

Also available **ReportEase Plus** for Windows featuring a **graphic form editor with line/box drawing, colors/shades, print preview**, etc. (\$419)

Spell Time

Spell Time consists of a dictionary (over 100K words) and the routines to access the dictionary. It also includes an application specific dictionary, and a user dictionary. The routine will suggest alternatives for a misspelled word. Highly optimized: 400 to 500 words/sec on a 33 Mhz 386 computer. An interface with TE included. Includes the complete 'C' source code. Specify **DOS, Windows or OS2-PM**. (\$369)

ChartPro

This Windows' DLL draws bar, pie, line, area, xyz point chart and hilo presentation graphs in unlimited styles. The output can be sent to a window, printer or a bitmap. Features 3d rotation and dialog boxes to edit chart parameters. Includes the 'C' source code. (\$379).

Sub Systems, Inc.

1-800-447-6819

Fax: 1-617-438-0311

159 Main Street, #8C, Stoneham, MA 02180, (617)-438-8901

can name all known VxD services by name and address, start and end of VxD objects and, particularly important for this article, all VxD Control procedures. Even debuggers such as WDEB386 don't provide as complete a map as VXD-QUERY. Since VXDQUERY is a commercial program of mine, source code is not provided; however, I'm making a special version available electronically for DDJ readers; see page 3.

To use VXDQUERY, another VxD simply loads an address into the EDI register and calls *VxdQuery_Address_To_VxD_Name*. The service returns the name of the VxD plus the closest known procedure that precedes the specified address. If you want to spy on the usage of a VxD call, as I did with *_AddInstanceltem*, write a VxD that uses the documented VMM function *Hook_Device_Service* to intercept the service at *Sys_Critical_Init* time, and collect the address of the callers plus any related parameters. You may need to initialize right after VXDQUERY. During *Init_Complete* or later, you can call VXDQUERY to translate your addresses into VxD names. Finally, your VxD should export a V86 or PM API to make your results public to the non-32-bit world (that is, to a program similar to INSTWALK).

Figure 2 (from LISTWALK) shows all instanced data with their linear addresses. At *Sys_Critical_Init* time, for example, VKD instances 28h bytes at 415h, one byte at 471h, 4 bytes at 480h, and 0Bh bytes at 496h. To make any sense of these numbers, you need to consult a reference that describes standard PC absolute-memory locations. A good source is the file MEMORY.LST included with Ralf Brown's "Interrupt List" (see IBMPRO library 5 on CompuServe); another source is *Undocumented PC* by Frank van Gilluwe (Addison-Wesley, 1994). In the VKD example, the 28h bytes at 415h include the keyboard buffer and head and tail pointer, 471h is the Ctrl-Break flag, 480h points to the keyboard buffer, and 496h includes keyboard-status bytes. It makes sense that VKD wants to instance these portions of the BIOS data area.

Of particular interest are the final calls to *_AddInstanceltem* in Figure 2. VMM made them from an internal routine called *Create_Int_2F_Inst_Table*; Figure 1 explains where this name comes from. This is the VMM code that processes the *Win386_Startup_Info_Struc* chain passed back from INT 2Fh AX=1605h; this documented structure includes an *SIS_Instance_Data_Ptr* field listing items that software loaded before Windows (such as DOSKEY) wants instanced.

VXDLOAD.EXE determines the names of TSRs which supply a Win386_SIS.

LISTINST.386 gets a pointer to the list of all SISs on entry in the *Sys_Critical_Init* procedure in EDX, because VXDLOAD fills the *Win386_SIS* for LISTINST.386 with a pointer to its reference data.

VMM first lets all virtual devices instance their data and then calls *Create_Int_2F_Inst_Table* to convert the instance structures provided by DOS TSRs via their *Win386_SIS* to the VMM (IDM) instance structures. The result is a doubly linked list of instance-data structures. You can walk the chain during *Init_Complete* with the *InstLinkF* and *InstLinkB* pointers. After the linked list is complete, the instance-description buffer will be initialized with the data from the linked list. Finally, the so-called "snapshot" is taken in order to save the start-up-time values of the instanced data in the instance snap buffer. If a new VM is created, the IDM creates a VMx instance buffer (where x is the VM ID) and copies the contents of the instance snapshot buffer into it.

There is one problem with this technique of hooking *_AddInstanceltem* to build up a picture of the instance description buffer. As noted, LISTINST.386 hooks *_AddInstanceltem* as early as possible: at *Sys_Critical_Init* time, right after VMM, and before other VxDs. Unfortunately, this is too late to intercept the *_AddInstanceltem* calls that VMM makes to support the LOCALTSRS= statement. However, LISTINST is still able to find these instance items by following the doubly linked list of *InstDataStrucs*. VMM will instance the entire program, excepting its environment segment.

It would also be useful to determine the ownership of the PG_INSTANCE pages. Again, only one VM at a time can own a PG_INSTANCE page. LISTINST.386 provides the array to LISTWALK. Type LISTWALK -p to dump the instance page-ownership array and the instance-description buffer. The output is more interesting if, immediately after INSTWALK starts, you switch to another session or compile something in the background. In this case, the instance pages are owned by different VMs.

Not Just Spying

What can you do with this information? In addition to a better understanding of how instance data works and what sorts of data must be instanced, the information presented here might lead to some interesting techniques for accessing instance data without causing page faults. This will be taken up in a future article.

DDJ

(Listing begins on page 146.)

To vote for your favorite article, circle inquiry no. 14.

Indexes of
**Dr. Dobb's
JOURNAL**

1982-1993

- *Cumulative Subject and Author Indexes covering the years 1982 through 1993, bound in ONE volume.*
- *An 8"×11" paperback with 136 pages and 9300 entries.*
- *Detailed coverage of all editorial material, including reviews, letters, tables, sidebars, bug reports, corrections, etc.*
- *Find that article, program, product review, algorithm, or routine (by language or type) in seconds, or follow the threads of a technical conversation.*
- *Hypertext version available on 3.5" disk (DOS).*

BOOK: \$21.95 ppd
(Canada \$22.85 U.S.)

DISK: \$14.95 ppd
(Canada \$15.70 U.S.)
VISA/MC

Previous purchasers of the diskette version may purchase the latest edition for \$10.00 (Canada: \$10.85 U.S.).

Voice/Fax:
804-977-7015

Stephen Bach
1411 A Short 18th Street
Charlottesville, VA 22902-5402

Please specify 5" or 3" when ordering. For shipping paperback edition via first class mail add \$1.75 (Canada \$2.00 U.S.).



3 LOOK INTO CLEAR ADVANTAGES OF USING SELECT OMT™

1.

ENHANCES SOFTWARE DESIGN

SELECT OMT puts your software design into clear focus. This professional Windows-based design tool uses a powerful object oriented technique, Rumbaugh's OMT, to capture the essence of your software design on screen. Based on this well-defined and easy-to-understand method, SELECT OMT enables you to graphically portray the class structures and relationships, the system states, and the functional structure of your entire system. The three model viewpoints—Object, Dynamic, and Functional—make your designs easy to read, validate and improve.

2.

PROVIDES A DETAILED BLUEPRINT

Just as a blueprint enables a builder to validate the structural integrity of a house before pounding the first nail, SELECT OMT enables you to produce and validate a "software blueprint" of your design before writing a line of code. This blueprint will help prevent system errors, ensure commonality between project members, and facilitate maintenance. SELECT OMT goes beyond a traditional builder's blueprint, translating your design directly into C++ framework code, suitable for any compiler—saving you hours of valuable time.

3.

SHARPENS YOUR IMAGE

SELECT OMT improves your ability to present your design to management and end users by producing high quality documentation and a graphical representation of your idea. SELECT OMT utilizes your existing software investment by integrating with IDEs like Borland's and Microsoft's, impressing financial management too.

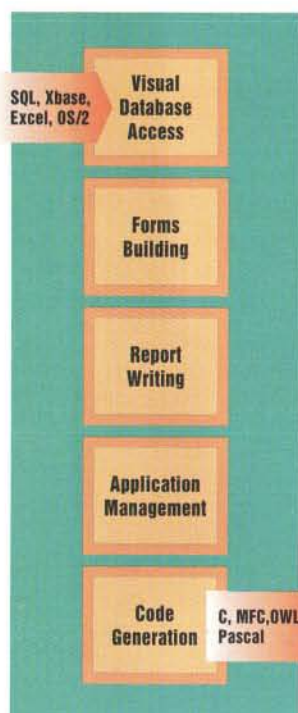
**SELECT
OMT**

Call 800-577-6633 to learn about SELECT OMT's other advantages—including its price.

Select Software Tools
1526 Brookhollow, Ste. 84, Santa Ana, CA 92705 (714) 957-6633
England 44-242-226553; Benelux 31-5270-16633; Germany 49-89-3600-8213;
France 331-4768-8080; Switzerland 41-1740-4105; Israel 972-3964-7136
All products referred are trademarks or registered trademarks of their respective holders.

Client/server with a **VISUAL** a d v a n t a g e

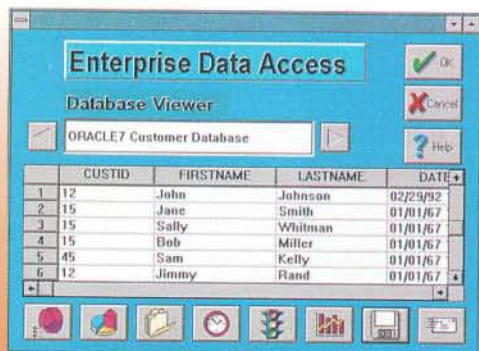
The ProtoGen+ Client/Server Suite. Visual tools that harness the power and productivity of C and C++.



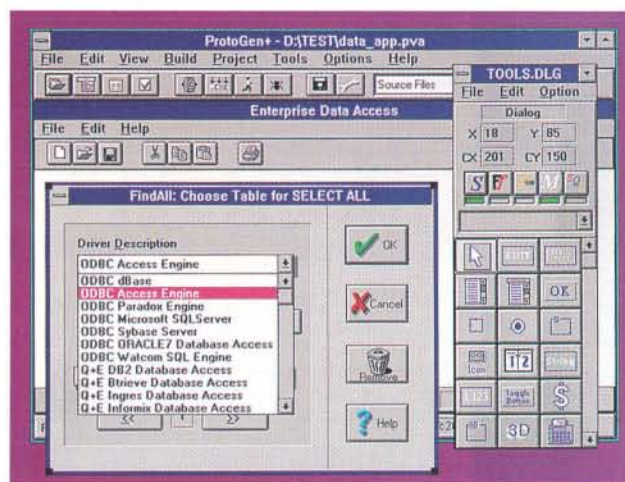
• The ProtoGen+ Client/Server Suite is a complete visual development environment that features multiple database access and quality code generation.

No pain, no gain. That's what most client/server tool vendors would like you to believe. Because their 4GL and interpreted language tools force you to learn proprietary languages and lock you into limited environments.

The ProtoGen+ Client/Server Suite however, harnesses the power and robustness of C and C++, the world's most popular development languages. So Windows developers—and former mainframe programmers learning Windows—can build mission-critical client/server applications with easy-to-use visual tools.



• Windows applications built with the ProtoGen+ Client/Server Suite deliver direct graphical access to enterprise SQL databases.



• The ProtoGen+ Client/Server Suite provides a powerful workbench that lets you use point-and-click tools to achieve the exact look and feel you want.

This visual, integrated best-of-breed toolset delivers unprecedented speed and productivity in an open architecture. Including database access. Forms building. Report writing. Application management. And quality code generation in C, MFC, OWL and Pascal.

Over 100,000 developers use ProtoView visual tools, including Delta, Merrill Lynch, AT&T and Conrail, for the mission-critical applications they

need in today's complex client/server environments.

So unless you enjoy 4GL pain, get the Visual Development Edge. You have everything to gain.

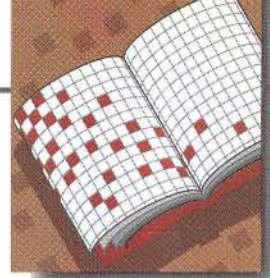
Call today for your
free Information Kit.
1-800-231-8588
Ext. 10


PROTOVIEW™
The Visual Development Edge™

Fax: (908) 329-8624 Outside the U.S., call (908) 329-8588

All products named are trademarks of their respective companies. ©1994 ProtoView Development

CIRCLE NO. 432 ON READER SERVICE CARD



A Clear Look Through Bleary Eyes at Two Books on Algorithms

Tom Ochs

Algorithm: A set of well-defined rules for the solution of a problem in a finite number of steps.

Books on algorithms are important tools for the professional software developer. However, as can be seen from the definition, the breadth of issues covered under the heading of algorithms can make the selection of the proper book difficult. Topics can cover numerical applications, business applications, data structures, searching, sorting, optimization, and many others. Some generic characteristics seen in algorithm-related books (ARBs) include: How algorithms are designed, why a particular algorithm is chosen, what measures are used to assess algorithm effectiveness, construction considerations, instances of the algorithms, application examples, test cases, reliability issues, comparisons with other algorithms, and data-structure dependence. ARBs also have an associated level of difficulty that can range from introductory through intermediate and advanced, to specialized-advanced (where you, the author, and three others in the world are interested in the topic). The tone can vary from practical to academic, and the presentation, from well written to just plain poor quality.

Clearly, the selection of a book on algorithms is situational, depending on your needs of the moment. A lot of books are collecting dust on my bookshelves because their characteristics don't meet my current needs. We should take the opportunity to use our analytical skills to determine our needs and then compare those needs to the characteristics of the available books. This can help make our investments in

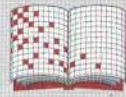
Tom is a consultant specializing in the integration of modern software-development methods into technical organizations. He has over 15 years experience as a research scientist, has written a commercial numerical package, and is a registered mechanical engineer living in Albany, Oregon. Tom can be contacted on CompuServe at 70511,652.



Programming Classics: Implementing the World's Best Algorithms

Ian Oliver

Prentice Hall, 1993, 386 pp. \$38.00
ISBN 0-13-100413-1



Algorithms from P to NP, Volume I: Design and Efficiency

B.M.E. Moret and H.D. Shapiro
Benjamin/Cummings Publishing,
1991, 576 pp. \$41.95
ISBN 0-8053-8008-6

time and money work for us. To supplement your needs assessment, here is my analysis of two relatively new books. Like movie critics who rate films from one to four stars, I will use a π rating, with π being a book that has little to offer, $\pi\pi$ being a book with marginal impact, $\pi\pi\pi$ having significant contribution, and $\pi\pi\pi\pi$ being a book that *must* reside on the serious developer's shelf.

Programming Classics

Any book that purports to be "detailing the best algorithms ever devised for a wide range of practical problems..." has a huge challenge ahead of it just to live up to the propaganda on the jacket. Unfortunately, *Programming Classics: Implementing the World's Best Algorithms*, by Ian Oliver, falls far short of the hype. Even though it does cover a wide selection of applications, the coverage is spotty, sometimes shallow, and generally incomplete. In trying to limit the complexity of the presentation, Oliver has also limited its usefulness. On numerous occasions he resorts to hand waving such as: "...is beyond the scope of this book...", "We will not analyze in detail...", "Do not use this algorithm unless you know what you are doing," and "Given the mathematical sophistication

needed for dealing with eigenvalues, no discussion of the reasons why the algorithm works will be given." Oliver has mistakenly tried to keep the presentation at an introductory level while introducing intermediate-level algorithms and concepts.

The lack of detailed discussion on the theory of operation of many of these algorithms leaves you to accept Oliver's choice for the implementation based on faith alone. If you have to modify, debug, or optimize the functions, the presentation in this book is generally inadequate. The inconsistency in the amount of detail is illustrated by the adequate coverage of sorting methods, including performance comparisons and application-specific suggestions, while the section on arithmetic is devoid of explanation.

In the poorly explained section on arithmetic, rational methods are introduced and a warning is given:

"...the methods will fail when integer overflow occurs. For certain practical applications it will be necessary to implement the algorithms in multiple precision arithmetic. The algorithms for multiple precision calculations are beyond the scope of this book."

What Oliver doesn't say is that the methods *generally* fail after only a few operations due to overflow, and the use of greatest-common-denominator (GCD) reduction is only temporarily effective at preventing the overflow. His presentation also skirts the fact that this implementation only works for toy problems if multiple-precision arithmetic is not used.

The author uses his own generic language, reminiscent of Ada, to define his "code" examples. His intent was to produce a broadly targeted representation that was language independent. Instead, it will be difficult to translate some of the code to older languages such as Fortran, Cobol, or C. The example code exhibits problems with initialization, typing, character/byte access, parameter passing, memory usage, and other implementation issues. Since these issues

are addressed in a generic way, it is almost assured that few real languages will come close to mapping transparently to his representations, forcing the users to modify their implementations without a clear understanding of the algorithm-design issues. If Oliver was serious about producing reliable code for readers to use directly, he should have chosen specific target languages so the syntax questions could have been dealt with in his implementations.

I rate *Programming Classics* π , for poor execution of a fundamentally good concept, useful only as the first place to look to find references to more complete explanations of the problems to be solved. While this book could be useful to experienced designers looking for a reference that gives terse overviews and points to other sources for details, it will be a hazard for the inexperienced designer looking for a quick method to solve a poorly understood problem.

Algorithms from P to NP

Algorithms from P to NP is a careful, academic text designed for graduate students, upper-level undergraduate students, and computing professionals prepared to use rigorous mathematical analysis in problem solving. If you aren't

comfortable with set notation, discrete mathematics, data structures, calculus, and algebraic expression of problems—pick another book. *Algorithms from P to NP, Volume I, Design and Efficiency*, by Moret and Shapiro, is clearly designed as an advanced textbook to be used in a classroom setting with an instructor and does an excellent job in that context. It also serves as a good refresher and reference for those who have been through similar advanced courses. I particularly liked the presentation and felt that Moret and Shapiro did a good job of leading the student through the solution process; however, it is industrial-strength analysis and not for the faint-of-heart. But it is worth the effort. The authors expect you to recognize standard algebraic notation, but introduce specific concepts with which you might not be familiar—a spanning tree, O-notation, generating functions, and directed graphs. The exercises range from simple examples to thesis-level assignments.

Algorithms from P to NP concentrates on combinatorial optimization problems and takes a thorough, depth-overbreadth approach. Moret and Shapiro start with several traditional problems such as the knapsack problem (filling a knapsack with the optimal mix of things

for a camping trip) and the traveling salesperson problem (traveling through a series of cities while optimizing time or distance). These problems are revisited throughout the book in generic instances as the problem-solving approach is modified and expanded to encompass extensions of the problems. You're given more tools to deal with increasingly difficult examples of the problems as the book progresses.

The reference to a "stack of punch cards," which most graduate students have never seen, dates the origin of some of the examples while demonstrating the timelessness of the problems. Throughout the book, there is just enough nerd humor (my favorite kind) to liven up a graduate course in algorithms. The basic approach of the book is one that I am comfortable with: "The study of algorithms cannot be dissociated from the study of problems." Their approach is to start with problem solving and then show how the solutions map naturally into an algorithm for the effective solution of the problems. They spend time reviewing methods for assessing algorithm run time, but they deal only peripherally with the concept of reliability. This limited discussion of reliability is probably related to the focus on combinatorial problems, as opposed to numerical issues. *Algorithms from P to NP* discusses not only the theoretical, asymptotic behavior of the algorithms, but also the application and implementation issues that impact performance. The language used for example code is Pascal, and the code examples have been used and tested in classroom situations.

The name of the book reflects the concentration in this volume on problems that have solutions which, as a worst case, require "Polynomial time" ($O(N^k)$, where N is the number of items and k is some constant) for their completion. These are represented as P-problems. The second volume deals with NP-complete problems (problems for which no solution has been found that can be completed in polynomial time). NP-complete problems are an ongoing subject of research, and are generally solved by approximation methods.

I rate this book $\pi\pi\pi$, for concise, clear explanations of problem-solving issues. A serious textbook for serious study of combinatorial issues. Don't pick this book up for light reading!

(For reviews of 14 ARBs dealing with numerical issues, refer to my "Building Blocks" column in the former *Computer Language* magazine, November, 1992.)

DDJ



NEW
HIGH
SPEED
RISC
I/O

Call us today at telephone 519/836/1291 or facsimile 519/836/4878 to find out more about the Intellicon-NT960 and other Connect Tech communication products

Intellicon-NT960

a RISC based multi-port I/O subsystem for serial communications

Performance

- ★ An on-board Intel i960 32 bit RISC processor off-loads the serial I/O task from the main CPU
- ★ Offers up to 2MB of on-board dynamic RAM for data storage
- ★ Offers up to 256KB of on-board dual-ported RAM and 512KB Flash EEPROM for data storage and custom application programs
- ★ RISC-like quad UARTS with 24 bytes of FIFO per channel provide high speed data communications up to 115K baud on each port

Hardware Flexibility

- ★ The Intellicon-NT960 subsystem includes a NT960 host adapter, the ACM/16 external communications module, software drivers and manuals
- ★ Provides 16 to 128 asynchronous serial ports from one PC slot
- ★ Accommodates a total

- of 512 serial ports with 4 host adapters in a system
- ★ The ACM/16 module has both RJ45 and DB25 connectors
- ★ Offers optional on-board realtime clock and optional on-board battery backup for the dual-ported RAM
- ★ Supports 80286, 80386 and 80486 AT bus architecture

Software Flexibility

- ★ Development tools available for custom development
- ★ Software support for UNIX, XENIX, QNX, DOS

Reliability

Since 1985 Connect Tech's products have become known for their unsurpassed quality and reliability. This tradition of providing flexible cost effective solutions is based on Connect Tech's expertise in the design and manufacture of communication hardware and software. At Connect Tech we meet your objectives — by design.



Connect Tech Inc.
'Meeting your objectives...by design'

727 Speedvale Ave. W.
Guelph, ON N1K 1E6

Tel: 519-836-1291
Fax: 519-836-4878

CIRCLE NO. 771 ON READER SERVICE CARD

To vote for your favorite article, circle inquiry no. 15.

RS#	Advertiser Name	Page #	RS#	Advertiser Name	Page #
75	Abraxas Software.....	87	914	Paxcell Group.....	150
858	AccuSoft.....	30	874	Performance Computing.....	113
912	Algorithms Corporation.....	24	214	Periscope Company, Inc.....	61
394	Annabooks.....	68	882	Periscope Company, Inc.....	97
915	Applied Data Systems.....	99	186	Phar Lap Software, Inc.....	41
849	Archimedes Software.....	20	825	Phase 3 Software.....	7
89	Austin Code Works.....	149	539	PKware, Inc.....	4
850	Auto-Data Systems.....	95	877	Poet Software.....	29
1215	Bayfront Technologies.....	138	321	Popkin Software & Systems, Inc.....	23
493	Borland International.....	33	898	Powerline Software.....	62
95	Borland International.....	C4	*	PowerSoft.....	48-49
851	Bristol Technology.....	25	826	Premia.....	65
1212	Bristol Technology.....	137	**	Product Debut.....	137-139
247	Burton Systems Software.....	101	**	Programmer's Marketplace.....	140-142
1225	Byte Craft Ltd.....	139	195	Programmer's Paradise.....	14-15
824	Byte Dynamics.....	70	432	ProtoView Development.....	132
908	C User's Journal.....	117	919	Quarterdeck Office Systems.....	83
742	CenterLine.....	76	501	Quarterdeck Office Systems.....	85
803	Computer Associates.....	11	1224	Raleigh Group International.....	139
771	Connect Tech, Inc.....	134	785	Rational.....	53
683	Consensus Corporation.....	113	520	Realtime Control.....	78
712	DABIWA.....	91	439	Rogue Wave & Associates.....	123
310	Datalight.....	66	814	RSA Data Security.....	31
901	DataPak Software, Inc.....	16	1220	Ryle Design.....	138
835	Diab Data.....	71	1223	SAX Software.....	139
*	Digital Equipment Corporation.....	47	854	Select Software Tools, Inc.....	131
838	Distinct Corporation.....	74	892	Sense 8 Corporation.....	145
*	DDJ Software Careers.....	96A/B	213	Sequiter Software, Inc.....	C3
*	DDJ CD ROM.....	135	888	Sietec Open Systems.....	73
*	Embedded Systems Conference.....	104	829	Signum Systems.....	70
489	Evergreen CASE Tools.....	63	216	SLR Systems.....	96
128	FairCom.....	90	801	Software Solutions.....	66
456	Fleming Software.....	78	782	Star Division.....	42-43
1226	Franz, Inc.....	139	903	SUNPRO.....	8-9
285	Genus Microprogramming.....	84	264	Sub Systems, Inc.....	130
1222	Geodysey.....	139	131	Symantec Corp.....	5
*	Gimpel Software.....	28	237	Symmetry Group.....	50
*	Gimpel Software.....	128	*	Tab Books, Inc.....	81
479	Greenleaf Software.....	124	1216	Technosoft.....	138
909	HPI.....	98	1211	Ted Gruber Software.....	137
737	IBM Personal Software Products.....	51	897	Thinking Software, Inc.....	146
*	IBM Personal Software Products.....	75	1221	Thompson Automation.....	139
*	IBM Personal Software Products.....	93	1213	Time Arts.....	137
822	ICS-Xhibition.....	88A/B	240	TurboPower Software.....	118
913	Ilog.....	12	609	Turning Point Software.....	119
760	Incredible Technologies.....	70	*	V Communications, Inc.....	54
1219	Info Magic.....	138	763	Van Nostrand Reinhold.....	129
*	Inmark Development Corporation.....	27	754	Visix Software, Inc.....	36-37
895	International Systems Design.....	80	698	Watcom, Inc.....	21
190	Intersolv.....	17	655	XVT Software, Inc.....	35
262	Lahey Computer Systems, Inc.....	46	329	Zylab Corporation.....	117
756	LEAD Technology, Inc.....	144			
365	Liant Software Corporation.....	108			
921	Lifeboat Publishing.....	94			
304	Lifeboat Publishing.....	114			
144	Lugaru Software Ltd.....	151			
910	M Technology Assoc. of North America.....	58			
1217	Machine Independent Software Corp.....	138			
918	MainSoft Corporation.....	39			
1214	Man Machine Interfaces.....	137			
900	MathSoft, Inc.....	79			
158	MetaWare Incorporated.....	13			
*	Microsoft Corporation.....	C2,1			
*	Microsoft Press Corporation.....	55			
845	MIS PRESS.....	110-111			
166	Mix Software.....	120			
*	MSJ.....	122			
805	NetManage.....	80			
170	Neural Ware, Inc.....	59			
602	Neuron Data Corp.....	57			
554	Nohau Corporation.....	69			
*	Nu-Mega Technologies.....	112A/B			
887	O'Reilly & Associates, Inc.....	88			
*	Object Design.....	77			
622	On Time Marketing.....	68			
599	Opus Software.....	96			
*	OS/2 World Conference.....	105			
1218	Pacific Softworks.....	138			
789	Paradigm Systems.....	67			

The index on this page is provided as a service to our readers. The publisher does not assume any liability for errors or omissions.

*The advertiser prefers to be contacted by phone; consult ad.

**See individual ads for RS#.

Advertising Sales Offices

Eastern Sales Manager

John Moon Mid-Atlantic/Southeast (508) 541-7425

Regional Sales Managers

Mary Kay Hoal	New England	(707) 557-1608
Paul Miller	Northern California	(415) 358-9500
Eric Pitts	Midwest	(415) 358-9500
Lauren Sargeant	Southwest/Texas	(310) 472-4577
Stephanie Vandenburg	Northwest	(415) 358-9500

Associate Account Manager

Cheryl Canion Marketplace/Debut (415) 358-9500

PRODUCT DEBUT

Call Cheryl Canyon today at (415) 358-9500 to develop a customized advertising campaign tailored to your marketing needs. Turn the page for more products.



Available on our BBS!
Another excellent game written with
Fastgraph™

Devasoft has released an excellent preschooler's edutainment program:
Amy's Fun-2-3 Adventure!

Sure to be a hit among the younger set! Written with Fastgraph, of course.
Fastgraph Programmer's Graphic Library \$199

Unlock the secrets of graphics programming with the graphics library preferred by game programmers!

C • C++ • Pascal • Basic • Fortran

Ted Gruber Software
P.O. Box 13408
Las Vegas, NV 89112

Voice (702) 735-1980
FAX (702) 735-4603
BBS (702) 796-7134

Visa/MC/COD

Demos available

No royalties

CIRCLE NO. 1211 ON READER SERVICE CARD



Time Arts **Signature**
Graphics Class Library

The object-oriented framework
for all your future graphics
development.



Time Arts

800-959-0509
707-576-7722
Fax: 576-7731

1425 Corporate Center Parkway
Santa Rosa, CA 95407-5434

CIRCLE NO. 1213 ON READER SERVICE CARD

Offer the HyperHelp Advantage



and be in good company!

WordPerfect • AT&T
Computer Associates (20/20) • Sybase
Siemens • EDS • Swiss Bank
Merrill Lynch • Corel (CorelDRAW!)
SPSS • Lotus (Ami Pro)

HyperHelp™ - The Industry Standard,
On-line Help Facility for Motif

Bristol Technology Inc.

Ph: (203) 438-6969 Fax: (203) 438-5013 email: info@bristol.com

FTP a demo: bristol.com, demo directory

CIRCLE NO. 1212 ON READER SERVICE CARD

EOS VBX v1.0

Visual Basic Programmers can now
access all the power of the
Evolutionary Object System from a new
Genetic Algorithm Custom Control!

Both Visual Basic and C++ programmers can now create Genetic Algorithm applications faster than ever. This Visual Basic wrapper around EOS for C++ gives a wider spectrum of developers access to a technology that Alan Kay has said, "...is one of the greatest intellectual happenings in computer science."

EOS and **EOS VBX** provide all the essential ingredients for building robust GA based applications. These include 3 types of encodings, multiple chromosomes, many different genetic operators, several selection methods, several reproduction strategies, elitism, and much more than we can describe here! In addition both products provide access to Man Machine Interfaces' exclusive adaptive technology where your GA adapts to the problem as it solves it. Please call for more technical details.

Don't miss out on this limited time discount pricing!

EOS VBX \$139.00 EOS for C++ \$139.00

EOS Professional (EOS C++ & VBX) \$199.00

Also ask about The Financial GENeius Tool Kit
for building GA based Trading Systems!!!

Order Today: 516-249-4700 • VISA • MC • Check
Man Machine Interfaces, Inc.

555 Broad Hollow Rd. Suite 230 Melville NY 11747
516-249-4700 Fax: 516-420-4085

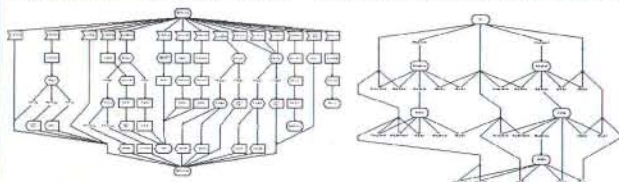
We are The Natural Selection for Genetic Algorithm Applications!

CIRCLE NO. 1214 ON READER SERVICE CARD

REAL-TIME CASE TOOLS

Bayfront's Computer-Aided Protocol Engineering (CAPE) Tools:

- Captures the protocol or state machine design
- Generates ANSI C Code targetable to any operating system or hardware platform
- Automatically draws state, state/event and CCITT SDL diagrams
- Rapidly creates a working model of the protocol
- Formalizes software state machine development
- Examples include the ISDN Q.931 protocol
- Currently in use internationally creating complex real-time, communications and client/server systems such as LAN Hubs, ATM and voice/data switches, distributed protocol applications and VSAT equipment



CAPE Tools for Windows \$550, Working model Demo disk \$45

Bayfront Technologies Inc, 1280 Bison B9-231, Newport Beach, CA 92660
For info: 714.436.0322 Fax: 714.436.1808

CIRCLE NO. 1215 ON READER SERVICE CARD

Locked Into PASCAL or BASIC? Wish Your Software Was In C?

Automatically Translate Your Old Code Into Readable and Maintainable ANSI C with the latest PASCAL and BASIC to C Translators.

Available For: Turbo Pascal, VAX Pascal/Basic, Microsoft Pascal/Basic

For More Information Call Now!

Technosoft (US): 815-397-3214
Technosoft (Europe): +44-264-781626

All Registered Trademarks Acknowledged

CIRCLE NO. 1216 ON READER SERVICE CARD

The Portable C-database System

Available with CQL

herCules

■ Supports Database, Memo, and Index formats of dBase III, dBase IV, Clipper & FoxBase

■ Available with the CQL Implementation of ANSI Level 2 SQL

■ Ciphering and Password Mechanisms

SUPPORTS

UNIX SYS V, BSD, OS/2, Windows 3.X and NT, DOS, Novell NLM.

Client/Server, ODBC, and library interfaces.

■ Royalty Free

■ Includes Shrouded Source

■ Compatible with any ANSI or K&R Compiler

■ Transaction Processing with Commit/Rollback

■ Documented Source Available

Machine Independent Software Corporation

491-B Carlisle Drive Herndon, VA 22070
703 435-0413 Fax: 437-8640 BBS: 437-8557

APIS Software

Bolongarostraße 113 D-65929 Frankfurt, Germany
Country Code 49 Tel: [0]69 30 39 06
Fax: [0]69 31 75 31

CIRCLE NO. 1217 ON READER SERVICE CARD

SNMP

Add SNMP Networking Protocols to your system design with:

FUSION Developer's Kit

- FUSION MIB 2 with Extensions
- T1 MIB, Ethernet Repeater MIB
- Modular Design Eases Porting
- Available in C source code
- Full Support, Training & Consulting

SNMP Ver. 2
Coming Soon



Pacific Softworks

For More Information Call

800-541-9508 or 805-484-2128 or Fax 805-484-3929

Also Available Embedded TCP/IP & Windows Developer Kits

CIRCLE NO. 1218 ON READER SERVICE CARD

Source Code on CD-ROM

CCITT Standards • RFC's & IEN's
Networking Tools & Utilities
386BSD • NETBSD • Free BSD • LINUX
X11RS • ALL GNU Sources
2nd Berkeley Networking Tape

CALL, FAX or send E-MAIL today!

InfoMagic

P.O.Box 708, Rocky Hill, NJ 08553-0708

(800) 800-6613 • (609) 683-5501 • Fax (609) 683-5502
info@infomagic.com

CIRCLE NO. 1219 ON READER SERVICE CARD

Graphics & Timing Tools

BGI SVGA Video Driver Toolkit

New Toolkit!

Provides Super VGA driver support at up to 1280x1024x256 for most popular SVGA cards using the Borland BGI graphics library. Support for graphics mode mouse and multiple video pages. Import/export popular bitmap file formats. Supports Borland DOS language compilers in real and protected mode. \$129.95 w/source.

BGI Font Toolkit

New Toolkit!

Complete replacement for the graphics text functions in the Borland BGI graphics library. Corrects bugs in BGI text functions, allows arbitrary rotation, bolding, underlining, and italicizing of BGI fonts. Supports additional bitmap font formats for filled font rendering. Works with any BGI video or hardcopy driver. Supports Borland DOS language compilers in real and protected mode. \$89.95 w/source.

BGI Printer Driver Toolkit

New Version!

Provides drivers for Borland's BGI graphics library to support a wide variety of printers, plotters, and bitmap file formats. Support for EMS/XMS. Print popular bitmap file formats. Extensive color device support. Not a screen dump - load our drivers with BGI's *initgraph* and get full output device resolution. Supports Borland DOS language compilers in real and protected mode. \$129.95 w/source.

BGI For Windows

Port DOS BGI Code to Windows

Gives you an interface to the Windows 3.x GDI compatible with Borland BGI graphics calls. Port your DOS BGI graphics code effortlessly to Windows. Full support for 256 color palettes, BGI stroke fonts, high resolution displays, and rasterized hardcopy. BGI extensions support TrueType and 24 bit color. Supports Borland Windows language compilers. \$129.95 w/source.

PC Timer Tools

Event Timing and Scheduling

Brings microsecond resolution timing to your DOS application with extensive functions for timers, delays, alarms, timer tick management, and thread scheduling. Ideal for execution profiling, data acquisition, and process control. Supports Borland DOS compilers, MSC/C++, Intel 386 CB, Zortech, \$89.95 w/source. PC Timer Objects for C++ and Turbo Pascal OOP - \$89.95 w/source.

All toolkits include

The Official Fine Print

complete source (in both C and Turbo Pascal), object/driver/DLL distribution license, and our 30 day "No Questions Asked" return policy. Add \$5 shipping USA, \$10 elsewhere. VISA, MasterCard, and American Express accepted. Our toolkit code is found in a wide variety of commercial and private applications in daily use by over 100,000 end users.

Ryle Design

Purveyors of Big Science since 1987

PO Box 22, Mt. Pleasant, MI 48804 USA
Voice/Fax: 517.773.0587 CIS: 73047,1765

Demos and spec sheets available on our BBS: 517.772.2393

CIRCLE NO. 1220 ON READER SERVICE CARD

Thompson Automation Software

Thompson Toolkit plus AWK Compiler

DOS & OS2 Programmers Make Life Easier with our Development Tools and Awk Prototyping Language

- Now 100% UNIX Korn Shell Compatible
- Toolkit allows you to set DOS Environment Variables and exceed DOS Command Line Limits
- New Awk Version 3.0 now supports Comma-Separated and Fixed-Field records

For full information contact:

Thompson Automation Software

5616 Jefferson • Portland, OR 97221

(503) 224-1639 • Fax (503) 224-3230 • 1-800-944-0139



CIRCLE NO. 1221 ON READER SERVICE CARD

VERSION CONTROL BREAKTHROUGH!

SourceSafe™

The only object oriented VCS for all platforms
(and less than 1/2 the price of PVCs!)

WINNER: *Microsoft Systems Journal's* competitive review

WINNER: *Computer Language's* Productivity Award

ENDORSED BY: Intel, Oracle, Motorola, American Airlines...

SourceSafe™ is:

- Project oriented version control
- A "FileManager" for your source code
- Cross-platform: DOS, WIN, OS/2, NT, UNIX, MAC...
- Incredibly easy to use

SourceSafe™ will:

- Install in 10 minutes
- Save you a minimum of 5 hours a week
- Track entire programs and projects
- NEVER lose a file

CALL NOW for a risk-free look at **SourceSafe™** and register to win an IBM ThinkPad™ (Limited to first 2,000)

1-800-364-5467 FAX: 1-919-848-0307

CIRCLE NO. 1224 ON READER SERVICE CARD

Hipparchus™ GIS Enabling Technology

Over 250 functions provide C developers with breakthrough GIS capabilities. Interface with any GUI and any DBMS. Ellipsoidal vector algebra enables seamless global coverage, lightning-fast spatial indexing, polygon overlay and more!

DOS/Windows Libraries, Application Prototyper, Tutorial (270 pp), Reference (340 pp): \$475. Most other platforms supported

Geodyssey Limited

500, 815 Eighth Avenue SW,
Calgary, Alberta, Canada T2P 3P2
403-234-9848 Fax 403-266-7117



CIRCLE NO. 1222 ON READER SERVICE CARD

Fu33-C

C preprocessor for fuzzy logic

- Integrated** – seamlessly combine fuzzy logic reliability with C power
- Portable** – use with most ANSI C compilers
- Extend C** – add membership functions, consequence functions and fuzzy logic control blocks to C
- Accessible** – easily call fuzzy logic functions from C

Byte Craft Limited (519) 888-6911

421 King St. N., Waterloo, Ont. N2J 4E4

CIRCLE NO. 1225 ON READER SERVICE CARD

Sax Comm Objects: makes Windows communications easier than ever



Modem database with over one hundred modems, easy and powerful functions.



Color Ansi, VT52, VT100 terminal emulation, scroll back and log file.



Low level control over handshaking, line settings, etc. Supports DigiBoard, Hayes ESP, etc.



Very fast background file transfers using X/Y/ZModem, Kermit, and CompuServe B+

What the experts say:

"For the C++ programmer, Sax Comm Objects is clearly the product of choice"

-- Windows Tech Journal

"The Sax Comm Control lets you plug in a sophisticated communications interface in a matter of minutes"

-- Borland International

30-day, money-back guarantee
1-800-MIKESAX



US: Phone 1-800-645-3729 Fax 913-832-8787
Int'l: Phone +32 37663264 Fax +32 37781405

CIRCLE NO. 1223 ON READER SERVICE CARD

We'll Tell You Just Once. Create Programs 10 Times Faster with Allegro CLPC.

Far more powerful than C++ or Smalltalk, Allegro CLPC is truly amazing. It offers you incremental compilation, memory management and a standardized library of over 700 functions and classes. Plus a built-in editor, debugger, browser and profiler.

Closely model the real world. Featuring CLOS, the Common Lisp Object System for advanced object-oriented programming, Allegro CLPC makes it easy to express difficult ideas. CLOS includes multiple inheritance, dynamic redefinition and a meta-object protocol for added extensibility.

Windows developers will love this development environment. You'll appreciate the native 32-bit compiler, access to the Windows API, and DLL support. Applications are portable to Unix workstations, and can be delivered royalty-free!



Call now and order Allegro CLPC for only \$595
—a \$400 savings!

OFFER EXPIRES MAY 31, 1994.



FRANZ INC.
FAX: (510) 548-8253

(800) 333-7260 ext. 93

CIRCLE NO. 1226 ON READER SERVICE CARD

MARKETPLACE

ASSEMBLY

Free Pentium Optimization Guide

Call or FAX for a copy of Quantasm's Pentium Optimization Guide. Used by itself or with ASMFLOW Professional, you can obtain amazing performance increases. ASMFLOW automatically produces flow charts, call trees register analysis, data x-ref, timing and more. Available for 80x86, 8051, 8085, 8096, Z-80, 6502, 68HC11 and 1750A. (\$199.95) Call for free demo and catalog of assembly, TSR and floating point libraries. (\$99.95 to \$299.95)

Quantasm Corp.
19672 Stevens Creek Blvd #307-D
Cupertino, CA 95014
800-765-8086 or 408-244-6826
FAX: 408-244-7268

CIRCLE NO. 1111 ON READER SERVICE CARD

BUSINESS OPPORTUNITIES

Earn \$4,000 Per Month From Your Home With A Computer!



FREE CBSI 486 Computer

Quit spending money on your computer and let it earn money for you. This is a proven turnkey business an individual or couple can run. If you purchase our software and business program, we will give you the computer and printer. If you already own a computer, you may receive a discount. Begin part-time and still retain the security of your present position. We will provide free, home office training. Financing available.

To receive free cassettes and color literature, call toll-free:

1-800-343-8014, ext. 1443
Or Write:

Computer Business Services, Inc.
CBSI Plaza, Ste. 1443, Sheridan, IN 46069

CIRCLE NO. 1112 ON READER SERVICE CARD

BOOKS

Earn Your High-Tech Degree While Working Full Time

New book *High-Technology Degree Alternatives* shows how to get your college degree without quitting your job, attending night school for years, or breaking your budget. All degree programs are accredited, and many do not require classroom attendance. Use VISA or M.C. \$21.95 + \$3.75 s/h. Professional Publication Inc., Dept. 809.

(800) 426-1178

CIRCLE NO. 1113 ON READER SERVICE CARD

BUG TRACKING

Bug Tracking Made Easy

Soffront® TRACK™ is a client-server based ready-to-use Bug-tracking and Technical Support system for Windows™. Using TRACK you can record a Bug, System, and Customer information, query, produce reports, notify other users, keep a log of change history, import, export, and more. Customize forms, fields, views and reports to meet specific needs of your organization.

Soffront Software
1806 Milmont Drive, Suite # 169, Milpitas, CA 95035
Call 1-800-SOF-FRONT/(408)263-2703
or Fax (408) 263-7452

CIRCLE NO. 1114 ON READER SERVICE CARD

C PROGRAMMING

Free Report on how you can Eliminate 87% of C Programming Errors in less than 2 hours!

This valuable report reveals secrets known by less than 5% of all professional programmers. These little known but powerful techniques prevent almost 90% of all design and maintenance errors with the guarantee that your compiled code will be virtually error free. Limited quantity available.

Logic Technologies
1 (619) 228-9653 FAX 1 (619) 369-1185
56089 29 Palms Hwy, Ste 254-DD, Yucca Valley, CA 92284

CIRCLE NO. 1115 ON READER SERVICE CARD

CD-ROM

640 MB OF SOURCE FOR LANGUAGES & OPERATING SYSTEMS FOR \$39.95

The LARGEST COLLECTION of source for compilers, function libraries, and docs for standard and research languages & OS ever assembled.

...Ada, Basic, C, C++, E, ET++, Lisp, Mach, Modula, Oberon, Pascal, Prolog, Rexx, Sather, Scheme, Self, Stimula, Tcl, Yacc, etc...

Language/OS CD-ROM	\$39.95
Other Titles Available	
Graphics - 1 (Applications w/ source)	\$24.95
Audio - 1 (Applications w/ source)	\$24.95
Multimedia - 1 (Applications w/ source)	\$49.95

Knowledge Media
436 Nunnely Rd., Ste B, Paradise, CA 95969, USA
1-800-78-CD-ROM
1-916-872-3826
VISA/MC/COD
Voice/FAX

CIRCLE NO. 1116 ON READER SERVICE CARD

CD-ROM

Your Best CDROMs!



Cica MS Windows CDROM	4000 new Windows programs - quarterly updates	\$29.95*
Giga Games CDROM	3000 hot Games for MSDOS and Windows	\$39.95*
Space and Astronomy CDROM	Thousands of NASA images and data files	\$39.95*
C User Group Library CDROM	A collection of user supported C source code	\$49.95*
Simtel MSDOS CDROM	Classic: 650 MB Shareware / Freeware for MSDOS	\$29.95*
QRZ! Ham Radio CDROM	PCC Callsign Database for Win & DOS + files	\$29.95*
Gifs Galore CDROM	6000 mostly 640x480 GIF Images in every category	\$39.95*
Gutenberg Project CDROM	Classic Literature and historical documents	\$39.95
Hobbes OS2 CDROM	600 MB always current Shareware/Freeware for OS/2	\$29.95*
Source Code CDROM	650 MB of Unix & DOS source code for programmers	\$39.95*
Internet Info CDROM	Thousands of computer, net, and internet documents	\$39.95
XII/GNU CDROM	X Windows and GNU software for Unix and SPARC	\$39.95
Amind Amiga CDROM	650 MB new Shareware/Freeware for Amiga	\$29.95*
GEMINI ADARI CDROM	650 MB 3000 Programs for Atari - also for BBS use	\$39.95*
Ada CDROM	Programming tools, Ada source code and docs	\$39.95*
Nova for NeXT CDROM	600 MB NeXT app's, etc., docs, etc.	\$59.95*
Nebula for NeXTSTEP Intel	600 MB NeXTSTEP Intel app's, docs, etc.	\$59.95*
Garbo MSDOSMac CDROM	MSDOS and Macintosh Shareware/Freeware	\$29.95*
Fractal Frayzy CDROM	2000 high resolution fractals - royalty free	\$39.95
FreeBSD CDROM	Berkeley BSD, 32bit OS for PC, w/ GNU & X11. Src	\$39.95
Linux CDROM	Yggdrasil Linux, 32bit OS for PC, w/ GNU & X11. Src	\$49.95
La Coleccion CDROM	MSDOS, OS/2, Windows. Spanish descriptions	\$39.95*
Sprite CDROM	Berkeley's experimental Distributed OS for Sun's	\$29.95
CDROM Caddies	Standard type, highest quality. Lifetime guaranteed!	\$ 4.95

1-800-786-9907

All our CDROMs are unconditionally guaranteed.

*Shareware programs require separate payment to authors if found useful.

Walnut Creek CDROM

4041 Pike Lane, Suite D-391 Concord, CA 94520
FAX: 1-510-674-0821 email: orders@cdrom.com

CIRCLE NO. 1117 ON READER SERVICE CARD

CONVERSION

Basic to C

BAS_C v6.0 translates BASICA/QBASIC/Quick BASIC source code to C source code automatically. The translated C source code is scoped, indented, structured, syntax error free and can be compiled Turbo/Borland/Microsoft/Quick C. Runtime C source code is available. 90% conversion ratio. From \$599. Call for non-MS Basic.

Gotoless Conversions

7105 Dee Cole Dr.
Colony, TX 75056

(214) 625-2323

BBS: (214) 625-6905 FAX: (214) 370-2612

CIRCLE NO. 1118 ON READER SERVICE CARD

DEVELOPMENT TOOLS

MENAI

Menai Corporation

DEVELOPMENT TOOLS

PARSING MADE EASY!

Validate input fields, parse configuration files, interpret data base queries, implement a macro or script language, design a "little language", build a compiler. Whatever your input requirements, produce more flexible, reliable, and maintainable programs faster with **AnaGram**, the powerful new parser generator built with programmers in mind. DOS, C, C++. Call for free trial copy.

AnaGram™ by Parsifal Software
P.O. Box 219, Wayland, MA 01778

800-879-2577 Voice/Fax 508-358-2564
CIS: 72603, 1763 jholland@world.std.com

CIRCLE NO. 1119 ON READER SERVICE CARD

EDUCATION

B.S. & M.S. In Computer Sciences

- ALL COURSES HOME STUDY.
- Approved for tuition reimbursement by leading corporations.
- Most courses interactive.
- Approved Ada course available.

**AMERICAN
INSTITUTE**

For free information call
1-800-767-2427 or write:
2101-DJ Magnolia Ave.
Suite 200 • B'ham, AL 35205

**COMPUTER
SCIENCES**

FORM SCAN

The Only Optical Form Recognition

Form Scan

Recognizer: Precise position and replicate
Text, Graphics and Fields with
all sizes and attributes.

Outputs: - C or Visual Basic, full
application with source
- Other popular form formats.

E-Data Link Inc. (713) 690-5710

CIRCLE NO. 1120 ON READER SERVICE CARD

FORTRAN



Fortran

**Indispensable Tools
For Serious Programmers!**

FORWARD \$399
• Save time and aggravation
• Find bugs faster
• Better understand large
programs

**FORTRAN DEVELOPMENT
TOOLS \$149**
• Find structure in your programs
• Clean up old, ugly Fortran code
programs

For DOS, Also for DOS protected mode: OS/2, Unix, and VMS
30 Day Money Back Guarantee!

Quibus
ENTERPRISES, INC.

3340 Marble Terrace • Colorado Springs, CO 80906 • (719) 527-1384

CIRCLE NO. 1121 ON READER SERVICE CARD

The SPINDRIFT Library Version 3.0

The Complete FORTRAN Programmer's Toolkit.
Create sophisticated state-of-the-art user interfaces for your FORTRAN programs. Spindrift's callable subroutines and functions let you

- Have data entry screens, dialog boxes, scrolling list boxes, pull-down menus, push buttons, help panels, etc. All with full mouse support.
- Much more! 281 subroutines and functions for keyboard, screen, mouse, DOS control, and hardware status.

Comprehensive demonstration program shows what you can do and how to do it.
Now shipping Version 3.0 with all new User Guide.

PRICE: 16 Bit Compilers - \$149 32 Bit Compilers - \$289

Spindrift Laboratories, Ltd.

116 South Harvard Avenue • Arlington Heights • IL 60005 • USA
Phone: (708) 255-6909 • FAX: (708) 255-6101

CIRCLE NO. 1122 ON READER SERVICE CARD

GRAPHICS LIBRARIES

IMAGE PROCESSING

Victor Library v 3.1. Powerful image processing & color reduction, bright/contrast, sharpen, outline, resize, overlay, matrix conv., etc. TIFF/BMP/JPEG/PCX/GIF/TGA/bin, works with large images without DOS extenders, EGA/VGA/SVGA/VESA, up to 16 million colors, LaserJet, ScanJet, for MS and Borland, C/C++. Source avail, no royal.

VICTOR LIBRARY for DOS \$195
VICTOR LIBRARY for WINDOWS \$295

Catenary Systems

470 Bellevue St. Louis, Mo 63119
Call/FAX (314) 962-7833

VISA/MC

COD

CIRCLE NO. 1123 ON READER SERVICE CARD

LIBRARIES

Windows Programmers - introducing... Data **DynaZIP** Compression for Microsoft Windows

Finally there's an intelligent and easy way to incorporate reading and writing of industry standard ZIP files into your Windows programs, without having to "shell" to DOS.

DynaZIP is a set of ROYALTY-FREE DLLs that give you a robust API for reading, testing, creating, writing, and updating ZIP files. Supports C/C++ and VB; includes source code for a high-quality Windows-based ZIP shell, comprehensive test/diagnostic tools, and full documentation/help system. Versions available for Windows 3.1 and Windows NT.

Now only \$249 w/30-day no-risk guarantee!
Call today, toll free: (800) 962-2949

Inner Media, Inc. • Hollis NH USA (603) 465-3216 fax (603) 465-7195

CIRCLE NO. 1124 ON READER SERVICE CARD

All mallocs Are NOT Equal

SmartHeap™ is an ANSI portable malloc and new for DOS, Windows, NT, OS/2, UNIX and other platforms. Proprietary algorithms deliver speed improvements up to 100X versus other commercial mallocs, especially for large heaps in the presence of virtual memory. SmartHeap localizes data structures in their own heaps. Plus it includes numerous fixed-size and handle-based APIs. Debugging facilities isolate leakage, overwrites, double-freeing, wild pointers, and many other errors to responsible file/line/pass count. Exceptionally reliable - used by a "who's who" of commercial software publishers. No royalties. Source available.

Call for FREE White Paper, Benchmarks,
and Error Reports

MICROQUILL SOFTWARE PUBLISHING, INC.
800-441-7822 / 206-595-8218 / FAX 206-595-8309
Compuserve:707512443 Internet:devtools@microquill.win.net

CIRCLE NO. 1125 ON READER SERVICE CARD

Communications Library

MSDOS and Windows 3.1 versions.

• Run 4 - 10 ports at once • Interrupt driven • 8250 - 16550 UART
• RTS - CTS flow control • Digiboard PC/X • 300 - 115200 baud
Includes example communications program in C with ASCII, XMODEM, and YMODEM protocols. Get source to library, printed User & Reference manuals, quarterly newsletter, one year BBS & phone support. Downloaded demo version from our BBS. Order the **Personal Communications Library** for DOS (PCL4C) \$65, Windows (PCL4W) \$95, or both \$135. Add \$3 S&H (\$6 overseas).

msc

**MarshallSoft
Computing, Inc.**

Post Office Box 4543
Huntsville AL 35815
205 881 4630 Voice / FAX
205 880 9748 BBS
"Reasonably Priced
Software Tools"

CIRCLE NO. 1125 ON READER SERVICE CARD

PGL - Printer Graphics Libraries

The PGL Toolkit is a set of easy to use libraries for generating device independent, high resolution graphics output on most popular printers. Includes full support for C/C++, Fortran, Basic, Pascal, Clipper, & Assembly plus 32-bit support for C and Fortran. NO ROYALTIES!

Only \$245 (includes all language support)
Source code available \$595!

AnSoft, Inc. Voice/FAX: (301) 470-2335

MARKETING

MARKET YOUR SOFTWARE

Whether you wish to license your software to an existing publisher, or form your own software company, we can help. We will provide legal and business services including product and market analysis, selection and solicitation of potential licensees, license negotiation, drafting and analysis of all contracts, incorporation, writing of business plans, and filing of financing applications. If your venture is a success we all profit, if it fails, you owe us nothing for our services.

Call or write today. No obligation. Confidential.
Do not send any software or ideas to us or anyone without a signed non-disclosure agreement!

Michael Norman Saleman, Attorney at Law
Rayman & Associates Consulting
15910 Ventura Blvd., Suite 801A, Encino, CA 91436
v: 818/379-8770 f: 818/379-8780 CIS: 71401,1653

CIRCLE NO. 1126 ON READER SERVICE CARD

ATTENTION: Application Developers, Consultants, and Small Software Publishers

Marketing can make you money,
get you new customers,
grow your business!

*Try Marketing Made Easy, the practical how-to-do
marketing newsletter for software professionals.*

Teaches how to create marketing pieces and programs.
Uses case studies, examples, and step-by-step guidelines.
Provides guidance in dealing with strategic
issues. Covers:

- Public Relations
- Market Research
- Direct Mail
- Tradeshows
- Upgrade Marketing
- And much more
- Pricing
- Data Sheets
- Brochures
- Press Kits
- Advertising



To order or for more information,
phone 1-206-641-3498 or
fax 1-206-644-0450

Special \$149/yr price for a limited time

No Marketing Experience Necessary!

100%
Money-back
guarantee

The Friedman Group
P.O. Box 40013
Bellevue, WA 98015-4013

CIRCLE NO. 1127 ON READER SERVICE CARD

NEURAL NETWORK

Neural Networks

As simple as a,b,c

- An easy-to-use applications package.
- Ideal as a Neural Net Introduction.
- Import/Export data from spreadsheets.
- Simple interface, no need for Ph.D. !!!
- Train & run your first neural net
2 minutes after installation!

Included with every package:
Normalization, Train File Maker,
Import, and Learn-2-Run utilities.
Example handbook and software.
Graphic network build and editing!

Neural Intelligence Ver 1.35.....\$115.00

(Add \$70.00 for Ver 1.35 runtime 'C' code generator.)

Neural Intelligence Integer.....\$290.00

Integer version for embedded systems development..

Cogito Software, Inc.

P.O. Box 451 Pine River, MN 56474

1-800-450-2001

Ask about demo disk
and booklet: "Basics
of Neural Systems"

CIRCLE NO. 1128 ON READER SERVICE CARD

M
A
R
K
E
T
P
L
A
C
E

NETWORK

Little Big LAN

Start small, grow as you do

- * Peer to peer LAN to 250 nodes
- * \$75 total software cost/LAN, not node!
- * Use Ethernet or Arcnet for speed
- * or Zero-slot via serial/parallel/modem
- * Share most anything in DOS or Windows
- * Needs only 40K of ram



Information Modes
Drawer F, Denton TX 76202
817-382-7407 FAX
1-800-628-7992 Orders

CIRCLE NO. 1129 ON READER SERVICE CARD

OBJECT ORIENTED

Why Can't Documenting Software be as Easy as Writing It!

It Can be by Using

BOCS

The only CASE Tool giving you the freedom and ease you've grown accustomed to when it comes to coding.

- Inheritance
- Consistency Checking
- Automatic Document Generation
- "Pretty Printing"
- System Level Features
- Traceability, Portable Libraries, 6 Models
- Extensive On line Help

And Lots More

Put an end to documentation drudgery! Only \$595
Includes both C++ and Smalltalk Code Generation

Call for a demo copy
Call 301-417-9884 or Fax 301-417-0021

Berard Software Engineering, Inc.
902 Wind River Ln., Suite 203, Gaithersburg, MD 20878
Email: info@bse.com

CIRCLE NO. 1130 ON READER SERVICE CARD

PROLOG TOOLS

Discover the Magic of Prolog...starting at \$49

DOS, 16 and 32-bit interpreter, \$49.
Interactive tutorial, \$75. Edinburgh-standard development system, \$149
Expert systems tutorial w/src, \$82.

FREE Catalog • MC/Visa Moneyback Guarantee

Amzit 40 Samuel Prescott • Stow, MA 01775
508/897-7332 fax -2784 amziod@world.std.com

CIRCLE NO. 1131 ON READER SERVICE CARD

SECURITY

FIGHT PIRACY PROTECT YOUR SOFTWARE

Since 1986, thousands of companies worldwide have chosen Az-Tech Software as partners in their fight against Software Pirates. Why? Because we offer you proven leaders:

EVERLOCK software Copy Protection Option Board Safe—Remote Registration New CPU LOCK-CD ROM LOCK and more EVERKEY HARDWARE LOCKS
Call for a FREE Demo
(800) 227-0644

Az-Tech Software, Inc.
201 East Franklin, Richmond, MO 64085
(816) 776-2700 Fax (816) 776-8398

CIRCLE NO. 1132 ON READER SERVICE CARD

SOFTWARE COPY PROTECTION

We build quality, security and affordability into our LTLOCK copy protection software. Evaluate, compare and see what we are offering:

- Installs easily on your IBM-PC Software
- Defeats all disk copy software
- Requires NO special devices or disks
- MSDOS 6 compatible
- Customizable options available

Evaluation disk (5 counts) \$25.
Starter kit (100 counts) \$145.
unlimited counts kit also available

L-Tech Corporation
2460 Lemoine Ave
Fort Lee, NJ 07024
Tel: 201-585-8820
Fax: 201-585-9022

CIRCLE NO. 1133 ON READER SERVICE CARD

SECURITY

CRYPKEY SOFTWARE LICENSING SYSTEM

"Hardware key like protection without the hardware key"

- CrypKey is a software protection tool, offering
- complete security from any disk copy program
- complete compatibility with any MS DOS or MSWINDOWS31 based machine
- complete invisibility—no disk key, no hardware key, less support calls
- instant disaster recovery

CrypKey is a sales tool, allowing you to sell your program

- by increments—enable the options the customer purchased
- by number of runs—e.g., sell 100 calculations for \$499.00
- by time period—e.g., lease or demo your program for 60 days

CrypKey uses a numeric key that can be transmitted by phone or fax. Sell your customers more options, more copies, more time or more runs instantly, just by making a telephone call. (Great for overseas customers or distributors.)

"NOW AVAILABLE FOR NETWORKS"

CRYPKEY IS PRODUCED BY KENONIC CONTROL—ENGINEERING AND SOFTWARE SINCE 1972.
Kenonic Controls Limited • 7175 - 12th Street South East • Calgary, Alberta, Canada T2H 2S6 • (403) 258-6200 • fax: (403) 258-6201

CIRCLE NO. 1134 ON READER SERVICE CARD

SOFTWARE SORTING

Opt-Tech Sort

High Performance Sort/Merge/Select Utility. Run from the command line or Call as a subroutine to your programs. Supports most languages and filetypes. Unlimited file size, multiple keys and much more! **New Version 5.0 includes More Speed and More Features!**

MS-DOS or Windows \$149 - OS/2 or Unix \$249

(702) 588-3737

Opt-Tech Data Processing
P.O. Box 678 • Zephyr Cove NV 89448

CIRCLE NO. 1135 ON READER SERVICE CARD

NO ROYALTIES

OmniSort Sorts, Merges, Selects, and Formats DOS, Btrieve and dBase files FAST. Runs as a stand-alone program or links with your Assembler, Basic, C, Cobol, Fortran & Pascal programs. Supports unlimited keys and four gigabyte files.

Regular \$129
Special \$99 until 4/30/94

803 786-7367

Omnware

501 Kinsler Road, Blythwood, SC 29016

CIRCLE NO. 1136 ON READER SERVICE CARD

SOURCE CODE ESCROW

SOURCE CODE ESCROW

It's finally here — A flexible source code escrow service for independent developers.

- Guarantee future support
- Enhance your professional image
- You retain copyright

Call today for a complete information package!



821 Fir Street • Sandpoint, ID 83864
1-800-569-7569 • FAX 208-265-4191

CIRCLE NO. 1137 ON READER SERVICE CARD

TOOLS

C and C++ DOCUMENTATION TOOLS

- **C-CALL (\$69)** Graphic-tree of caller/called function hierarchy, function/files index, function cross-reference.
- **C-CMT (\$69)** Creates/inserts comment-blocks (functions/identifiers used) for each function.
- **C-METRIC (\$59)** Calculates path complexity, counts lines with comments, code, and 'C' statements.
- **C-LIST (\$69)** Lists and action-diagrams, or reformats source into user selected standard formats.
- **C-REF (\$59)** Creates local/global/define/parameter cross-reference, C++ class hierarchy tree.
- **Special: C-DOC (\$199)** All 5 in 1 DOS (<15,000 lines)
- **C-DOC Professional (\$299)** DOS, OS2, Windows Processes 150,000+ lines, enhanced 3-ring binder.

SOFTWARE BLACKSMITHS INC.

6064 St Ives Way, Voice/Fax: (416) 858-4466
Mississauga, ONT Demos/BBS: (416) 858-1916
Canada L5N-4M1

CIRCLE NO. 1138 ON READER SERVICE CARD

UTILITIES

THE FILE ANALYST

Dump * Search & Replace * Display
ASCII * EBCDIC * xBase * Delimited
Analyze * Reformat * Translate * Split
Convert * Copy * Scan * Print
Verify * Undo Labels * And more!
20 menu functions to quickly identify, reveal, manipulate and view any data structure — no matter how complex. Perfect testing & import tool for uniform fields and records. Just fill in the blanks & run. No programming! \$169.00 Call for Demo.

212-779-7667

Ralph Garner Associates, Inc.
99 Madison Avenue, New York, NY 10016

CIRCLE NO. 1139 ON READER SERVICE CARD

AeSORT Sort Merge/Select ULTIMATE SORT SPEED

New Criteria for the words 'FAST sort'
Up to 12 TIMES faster than others, 10k size.
Versatile, Easy. DOS exe, batch or call subrn from popular languages. Mult keys & lengths, Asc/Dscn, output recs, keys or in's. Fixed, xbase, etc. Also powerful Merge/Select. \$159.

AeSORT Inc.

4070 E. Pueblo, Phoenix, AZ 85040
(602) 437-2867

CIRCLE NO. 1140 ON READER SERVICE CARD



DiskCloner Software disk duplication utility for Windows™

- DiskCloner is a super-fast software disk duplicator for Windows.
- Reads source disk images into memory - NO MORE DISK SWAPPING
- Saves disk images to hard disk for later use.
- Share disk images over a network for mass duplication.
- Automatically formats destination disks.
- Software Professionals: DiskCloner is ideal for making internal releases on demand.

To order, send a check or money order for \$29.95. Include \$3 for shipping in the US. Overseas orders add \$5 shipping. CA residents add tax.
North Beach Labs, 550 Union St. #402, San Francisco, CA 94133. Tel/Fax 415-693-0570

CIRCLE NO. 1141 ON READER SERVICE CARD

WINDOWS

Emacs for Windows

WinEmacs is a fully functional Windows 3.1 version of the industry standard program editor Gnu Emacs, version 19.3.

- WinEmacs HAS THESE EXTENDED FEATURES:**
- Separate buffers in different windows
 - Menu and drop-down menu bar
 - Multiple font size and type support
 - Cut and paste mouse support
 - Support for text and Binary files
 - Support for foreign keyboards
 - Clipboard support
 - Scroll bars
 - DDE and OLE support
 - Binds any arbitrary combination of a key and key modifiers to Emacs Lisp code

Contact **Pearl Software** at pearl@netcom.com (e-mail) (510) 273-9795 (voice) or (510) 839-9820 (fax) for more information. Supported version costs \$199.

CALL 1-800-WIN-EMAC

— WE ALSO PROVIDE EMACS CONSULTING SERVICES —

Pearl Software • 320 Lenox Avenue, Oakland, CA 94610

CIRCLE NO. 1142 ON READER SERVICE CARD



Why use a language from the 70's for applications of the 90's? Lingo source code is fully portable across different platforms. Contact us 24 hours a day for more information.

LINGO LANGUAGE • Phone 61(9) 485 7830 • Fax 61(9) 316 2873

CIRCLE NO. 1143 ON READER SERVICE CARD

Listing One (Text begins on page 121.)

```

(* ----- Search engine for IDENT program ----- *)
** search.pas -- Search engine for IDENT program
** Trie search algorithm
** Copyright (c) 1994 by Tom Swan. All rights reserved.
*)

unit Search;
INTERFACE
uses Common;

[ Return true if Ident is a Turbo Pascal reserved word ]
function IsReserved(Ident: IdentStr): Boolean;
IMPLEMENTATION
type
  ResWord = String[14];
  PResWordRec = ^ResWordRec;
  ResWordRec = record
    Word: ResWord;      [ Reserved word string ]
    Next: PResWordRec; [ List link field ]
  end;

var
  Index: array['a' .. 'z'] of PResWordRec;
[ Add word W to list at P ]
procedure AddList(var P: PResWordRec; var W: ResWord);
begin
  if (P <> nil) then
    AddList(P^.Next, W)
  else begin
    P := new(ResWordRec);
    if (P = nil) then
      begin
        Writeln('Out of memory');
        Halt;
      end;
    P^.Word := W;
    P^.Next := nil;
  end;
end;

[ Add word W to global Index ]
procedure AddWord(W: ResWord);
begin
  if Length(W) = 0 then exit;
  AddList(Index[W[1]], W);
end;

[ Initialize search engine variables ]
procedure Initialize;
var
  C: Char; [ Index[] array index ]
begin
  for C := 'a' to 'z' do
    Index[C] := nil;
    AddWord('and');
    AddWord('array');
    AddWord('asm');
    AddWord('begin');
    AddWord('case');
    AddWord('const');
    AddWord('constructor');
    AddWord('destructor');
    AddWord('div');
    AddWord('do');
    AddWord('downto');
    AddWord('else');
    AddWord('end');
    AddWord('export');
    AddWord('exports');
    AddWord('far');
    AddWord('file');
    AddWord('for');
    AddWord('function');
    AddWord('goto');
    AddWord('if');
    AddWord('implementation');
    AddWord('in');
    AddWord('inherited');
    AddWord('inline');
    AddWord('interface');
    AddWord('label');
    AddWord('library');
    AddWord('mod');
    AddWord('near');
    AddWord('nil');
    AddWord('not');
    AddWord('object');
    AddWord('of');
    AddWord('or');
    AddWord('packed');
    AddWord('private');
    AddWord('procedure');
    AddWord('program');
    AddWord('public');
    AddWord('record');
    AddWord('repeat');
    AddWord('set');
    AddWord('shl');
    AddWord('shr');
    AddWord('string');
    AddWord('then');
    AddWord('to');
    AddWord('type');
    AddWord('unit');
    AddWord('until');
    AddWord('uses');
    AddWord('var');
    AddWord('virtual');
    AddWord('while');

```

```

    AddWord('with');
    AddWord('xor');
  end;

[ Trie search algorithm ]
function IsReserved(Ident: IdentStr): Boolean;
var
  P: PResWordRec;
begin
  IsReserved := false;
  if Length(Ident) = 0 then exit;
  DownCase(Ident);
  P := Index[Ident[1]];
  while (P <> nil) do
    begin
      if P^.Word = Ident then
        begin
          IsReserved := true;
          exit;
        end;
      P := P^.Next;
    end;
  end;

begin
  Initialize;
end.

```

Listing Two

```

(* ----- Various constants, types, and subroutines ----- *)
** common.pas -- Various constants, types, and subroutines
** Copyright (c) 1994 by Tom Swan. All rights reserved.
*)

unit Common;
INTERFACE
const
  identStrLen = 64;
  digitSet = ['0' .. '9'];
  upperSet = ['A' .. 'Z'];
  lowerSet = ['a' .. 'z'];
  alphaSet = upperSet + lowerSet;
  identSet = alphaSet + digitSet + ['_'];
type
  IdentStr = String[identStrLen];
[ Return lowercase equivalent of Ch ]
function DnCase(Ch: Char): Char;
[ Convert all letters in identifier to lowercase ]
procedure DownCase(var Ident: IdentStr);
IMPLEMENTATION
[ Return lowercase equivalent of Ch ]
function DnCase(Ch: Char): Char;
begin
  if Ch in upperSet
    then Ch := Chr(Ord(Ch) + 32);
  DnCase := Ch;
end;

[ Convert all letters in identifier to lowercase ]
procedure DownCase(var Ident: IdentStr);
var
  I: Integer;
begin
  if Length(Ident) > 0 then
    for I := 1 to Length(Ident) do
      Ident[I] := DnCase(Ident[I]);
end;

begin
end.

```

Listing Three

```

(* ----- Convert key word identifiers in .PAS files ----- *)
** ident.pas -- Convert key word identifiers in .PAS files.
** Converts key words in Pascal listings to lowercase, and
** marks them for bold facing. Words are marked using the
** symbols <*> and *. For example, <begin> is interpreted as
** a bold faced "begin" key word. A word-processor macro could
** search for all <*> and * symbols in the resulting file and
** replace these with bold face on and off commands.
** Copyright (c) 1994 by Tom Swan. All rights reserved.
*)

{$X+} [ Enable "extended" syntax ]
program Ident;
uses Dos, Common, Search;
const
  bakExt = '.BAK'; [ Backup file extension ]
  tempExt = '.*$$$'; [ Temporary file extension ]
type
  PString = ^String;
  PListRec = ^TListRec;
  TListRec = record
    Path: PString;
    Next: PListRec;
  end;
  TState = (
    Reading, Chkcomment, Comment1, Comment2, Stopcomment,
    Stringing, Converting
  );
var
  FileSpec: ComStr; [ Files entered on command line ]
  Root: PListRec; [ File name list root pointer ]
  DelimitWords: Boolean; [ True to add <*> to reserved words ]
  CapitalizeWords: Boolean; [ True to capitalize non-keywords ]

```

(continued on page 144)

Listing Three (Listing continued, text begins on page 121.)

```

{ Return copy of a string }
function NewStr(S: String): PString;
var
  P: PString;
begin
  GetMem(P, Length(S) + 1);
  if (P <> nil) then
    PString(P)^ := S;
  NewStr := P;
end;
{ Return true if InF is successfully converted to OutF }
function ConvertIdents(var InF, OutF: Text): Boolean;
var
  Ch, PushedCh: Char;
  State: TState;
  Identifier: IdentStr;
  function GetCh(var G: Char): Char;
  begin
    if PushedCh <> #0 then
      begin
        C := PushedCh;
        PushedCh := #0;
      end
    else
      Read(InF, C);
      if (C = #13) or (C = #10) then
        begin
          if (C = #13) then
            WriteLn(OutF); { Start new line }
          C := #0; { Ignore new line characters }
        end;
        GetCh := C;
      end;
    procedure UngetCh(Ch: Char);
    begin
      PushedCh := Ch;
    end;
    procedure PutCh(Ch: Char);
    begin
      if Ch <> #0 then
        Write(OutF, Ch);
    end;
  begin
    PushedCh := #0; { No pushed character }
    State := Reading;
    while not eof(InF) do
      begin
        GetCh(Ch);
        case State of
          Reading:
            begin

```

```

              case Ch of
                '(': State := Chkcomment;
                '{': State := Comment1;
                '/*': State := Stringing;
              end;
              if Ch in alphaSet then
                begin
                  UngetCh(Ch);
                  State := Converting;
                end
              else
                PutCh(Ch);
              end;
            Chkcomment:
              if Ch = '*' then
                begin
                  PutCh(Ch);
                  State := Comment2;
                end
              else
                begin
                  UngetCh(Ch);
                  State := Reading;
                end;
            Comment1:
              begin
                PutCh(Ch);
                if Ch = ')' then
                  State := Reading;
                end;
            Comment2:
              begin
                PutCh(Ch);
                if Ch = '*' then
                  State := Stopcomment;
                end;
            Stopcomment:
              begin
                PutCh(Ch);
                if Ch = ')' then
                  State := Reading;
                else
                  State := Comment2;
                end;
            Stringing:
              begin
                PutCh(Ch);
                if Ch = '/*' then
                  State := Reading;
                end;
            Converting:
              begin
                Identifier := '';
                while Ch in identSet do
                  begin
                    Identifier := Identifier + Ch;
                    Read(InF, Ch) { Note: Don't call GetCh here! }
                  end;
                  if IsReserved(Identifier) then
                    begin
                      DownCase(Identifier);
                      if DelimitWords then
                        Identifier := '<*' + Identifier + '*>';
                      end
                    else
                      if CapIdentifiers and (Length(Identifier) > 0) then
                        Identifier[1] := UpCase(Identifier[1]);
                      Write(OutF, Identifier);
                      UngetCh(Ch);
                      State := Reading;
                    end;
                  end;
                if PushedCh <> #0 then { Write possible pushed last char that }
                  PutCh(Ch); { sets eof() to true. }
                ConvertIdents := true;
              end;
            { Convert one file specified in Path string }
            procedure ConvertOneFile(Path: PathStr);
            var
              Result: Integer;
              BakF, InF, OutF: Text;
              TempName, BakName: PathStr;
              Name: NameStr;
              Dir: DirStr;
              Ext: ExtStr;
            begin
              Write(Path);
              Assign(InF, Path);
              [{i-} Reset(InF); {i+}]
              if IoResult <> 0 then
                Writeln(' **Error opening file')
              else
                begin
                  FSplit(Path, Dir, Name, Ext);
                  TempName := Dir + Name + tempExt;
                  BakName := Dir + Name + bakExt;
                  Assign(OutF, TempName);
                  [{i-} Rewrite(OutF); {i+}]
                  if IoResult <> 0 then
                    Writeln(' **Error creating output file')
                  else
                    begin
                      if ConvertIdents(InF, OutF) then
                        begin
                          Close(InF);
                          Close(OutF);
                          Assign(BakF, BakName);
                          [{i-}]
                          Erase(BakF);
                          Result := IoResult; { Throw out IoResult }
                          Rename(InF, BakName);
                          Rename(OutF, Path);

```

Graphics POWER LEADTOOLS®

Add complete imaging to your apps!

Read, write, display,
print, process,
convert &
more
PLUS

*TGA
*DIB
*GIF
WMF

*TIF
*BMP
*PCX
EPS

COMPRESSION

CCITT 3 & 4, JPEG, LZW
& LEAD CMP

Fastest, tightest in the industry
(Now in Core!DRAW!)

78 routines, each with example code. And paste code from two free demo apps! LEADTOOLS kits from DOS to NT. Call now for full info & demo disk!

1-800-637-4699

LEAD Technologies, Inc.
8701 Mallard Creek Rd. • Charlotte, NC 28262
(V) 704-549-5532 • (F) 704-548-8161 *Export only. DD7


```

    {S1+}
    if IoResult <> 0 then
        Writeln(' **Error renaming files')
    else
        Writeln(' done')
    end else
        Writeln(' **Error processing files')
    end
end
end;

{ Convert files on global list at Root pointer }
procedure ConvertFiles(List: PListRec);
begin
    if List = nil then
        Writeln('No files specified')
    else
        while List <> nil do
            begin
                ConvertOneFile(List^.Path^);
                List := List^.Next
            end
        end;
    end;

    { Add file path to list }
    procedure ListFile(var List: PListRec; Path: PathStr);
    var
        P: PListRec;
    begin
        New(P);
        P^.Next := List;
        P^.Path := NewStr(Path);
        if P^.Path = nil then
            Dispose(P)
        else
            List := P
        end;
    end;

    { Create list of file names from FileSpec string }
    procedure ListFiles(var List: PListRec);
    var
        Sr: SearchRec;      { Directory search record }
        L: Integer;          { Length of Dir string }
        OldDir: DirStr;      { Old directory upon entry to procedure }
        Path: PathStr;       { Expanded file specification with path info }
        Dir: DirStr;         { Directory component of Path }
        Name: NameStr;       { File name component of Path }
        Ext: ExtStr;         { File extension component of Path }
    begin
        GetDir(0, OldDir);    { Save current path }
        Path := FExpand(FileSpec); { Add path info to file spec }
        FSplit(Path, Dir, Name, Ext); { Separate Path components }
        L := Length(Dir);     { Prepare to change directories }
        if L > 0 then
            begin
                if (Dir[L] = '\') and (L > 1) and (Dir[L-1] <> ':') then
                    Delete(Dir, L, 1); { Ensure that ChDir will work }
                ChDir(Dir);           { Change to location of file(s) }
            end;
        FindFirst(Path, 0, Sr); { Start file name search }
        while DosError = 0 do { Continue while files found }
            begin
                Path := FExpand(Sr.Name); { Expand to full path name }
                ListFile(List, Path); { Add path to list }
                FindNext(Sr); { Search for the next file }
            end;
        ChDir(OldDir)
    end;

    { Display instructions }
    procedure Instruct;
    begin
        Writeln('Use -b option to surround reserved words with');
        Writeln(' <*> for bold-facing in a word processor. ');
        Writeln('Use -c option to capitalize non-keyword identifiers. ');
        Writeln;
        Writeln('WARNING: After conversion with -b, the listing will');
        Writeln('not compile. Use -b ONLY on a copy of original files. ');
        Writeln;
        Writeln('ex. IDENT single.pas');
        Writeln('      IDENT -b one.pas two.pas');
        Writeln('      IDENT wild?.pas -b *.pas')
    end;

    { Main program initializations }
    procedure Initialize;
    begin
        Writeln;
        Writeln('IDENT -- (C) 1994 by Tom Swan');
        Writeln('Converts Pascal reserved words to lowercase. ');
        Writeln;
        Root := nil; { File name list is empty }
        DelimitWords := false; { Normally do not add <*> to words }
        CapIdentifiers := false; { Normally do not capitalize other ids }
    end;

    { Main program block }
    var
        I: Integer;
    begin
        Initialize;
        if ParamCount = 0 then
            Instruct
        else for I := 1 to ParamCount do
            begin
                FileSpec := ParamStr(I);
                if (FileSpec = '-b') or (FileSpec = '-B') then
                    DelimitWords := true
                else if (FileSpec = '-c') or (FileSpec = '-C') then
                    CapIdentifiers := true
                else begin
                    ListFiles(Root);
                end
            end
        end;
    end;
end;

```

```

ConvertFiles(Root)
end
end.

```

Listing Four

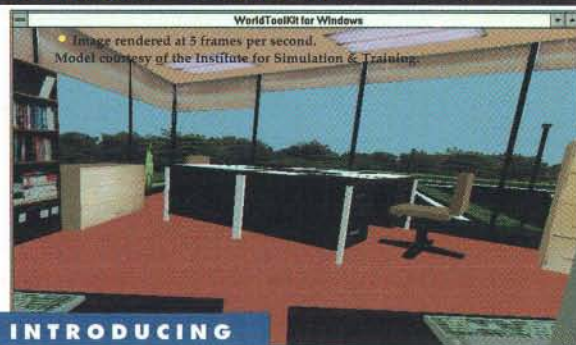
```

Sub MAIN
StartOfDocument
EditFind.Find = "<*>".WholeWord = 0, .MatchCase = 0, .Direction = 1, \
While EditFindFound()
EditClear
EditFind.Find = "<*>".WholeWord = 0, .MatchCase = 0, .Direction = 1, \
If Not EditFindFound() Then
Stop
End If
EditClear
WordLeft 1, 1
Bold 1
EditFind.Find = "<*>".WholeWord = 0, .MatchCase = 0, .Direction = 1, \
Wend
End Sub

```

End Listings

VIRTUAL REALITY *For Your Desktop and Beyond...*



WorldToolKit™ for Windows™

The power of real-time interactive 3D visualization has arrived on the desktop. With the new development environment, WorldToolKit™ for Windows, C programmers can create novel virtual world applications for home entertainment, architecture, CAD, education, and many other fields.

The core of the WorldToolKit development environment is a C library (DLL) of over 400 functions that dramatically simplifies the process of creating real-time interactive 3D simulations. WorldToolKit also ships with several

compiled applications allowing you to load in existing Autocad™ DXF or 3D Studio™ 3DS files and "fly" around them using a mouse.

FEATURES

- Supports Visual Basic and C/C++
- Real-time rendering of Gouraud-shaded and textured surfaces.
- Portability: DOS, Sun, SGI, DEC.
- Supports DDE and OLE.

Introductory price of \$795. 30 day MBG.

SENSE8 Corporation
PH: 415/ 331-6318 FX: 415/ 331-9148

WorldToolKit and SENSE8 are trademarks of SENSE8 Corporation, all other trademarks are the property of their respective companies.

Listing One (Text begins on page 125.)

```

;;; _AddInstanceItem hook from LISTINST.386

;;; from DDK VMM.INC
InstDataStruc struc
InstLinkF dd 0 ; linked list forward ptr
InstLinkB dd 0 ; linked list back ptr
InstLinkAddr dd ? ; Linear address of start of block
InstSize dd ? ; Size of block in bytes
InstType dd ? ; INDOS_Field or ALWAYS_Field -- ignored?
InstDataStruc ends

;;; from LISTINST.INC -- my InstData struct includes caller address
KM_InstData struc
AddInst_Caller dd ?
InstDataStruc { } ; from VMM.INC
KM_InstData ends

;;; from LISTINST.ASM
oldservice dd 0 ; return value from Hook_Device_Service
Inst_Struc_Ptr dd 0 ; InstLinkF from InstDataStruc
calladr dd 0 ; address of _AddInstanceItem caller
Data_Buf_Addr dd 0 ; created with _PageAllocate PG_SYS
Data_Buf_Size dd 0
Data_Buf_Handle dd 0
Inst_Data_Count dd 0 ; number of instance items seen so far

;;; from LISTINST.AM Sys_Critical_Init handler

;Instantiating of the first byte in the 1st MB in order to get all calls to
;_AddInstanceItem before ListInst_Sys_Critical_Init. The _AddInstanceItem
;service chains the InstDataStruc together to a sorted double linked list
;via InstLinkF and InstLinkB.
;If the LinkF field is -1, no other calls were made.
;If LinkF <> -1, then it represents a call to _AddInstanceItem caused by a
;system.ini - entry "LOCALTSRS= tsr.name". The VMM instances the whole TSR,
;the first 16 Byte represents the MCB of the PSP. So we can determine the name
;of the fully instantiated TSR.
;;; ...
mov KM_Instance.InstLinkAddr,0
mov KM_Instance.InstSize,1
mov KM_Instance.InstType,ALWAYS_FIELD
mov esi,offset32 KM_Instance
VMMcall _AddInstanceItem(esi,0)
cmp KM_Instance.InstLinkF,-1 ;any LOCALTSRS ?
je nocal
mov esi,KM_Instance.InstLinkF ;yes, get it
locbp: mov Inst_Struc_Ptr,esi
mov calladr,'LTSR'
call addinst ;add instance item to our list
mov esi,[esi.InstDataStruc.InstLinkF] ;get next InstDataStruc

```

```

cmp esi,-1 ;no more structs?
jne loclp

nocal:mov eax,_AddInstanceItem
mov esi,offset32 myhook
VMMcall Hook_Device_Service
mov [oldservice],esi
;;; ...
BeginProc Hooked_AddInstanceItem
; The AddInstanceItem Hook stores the callers address
; and the instance data pointer in the Inst_Data_Buf buffer.
myhook:
push ebp
mov ebp,esp
push [ebp+0ch] ; Flags
push [ebp+8] ; Instance Structure Pointer
push [ebp+8]
pop Inst_Struc_Ptr
push [ebp+4] ; get caller's return address!
pop calladr
call [oldservice] ; call original _AddInstanceItem
add esp,8
pop ebp
cmp eax,0 ; error in _AddInstanceItem ?
je exit
call addinst ; add Instance Item to our list
exit: ret
;*****
;addinst - adds an instance item to our list.
;INPUT: calladr - address of caller of _AddInstanceItem
; Inst_Struc_Ptr - address of InstDataStruc
;OUTPUT: hookerr = -1 - error growing Data_Buf
; hookerr = 0 - all O.K.
;*****
addinst:push edi
push esi
push ecx
push eax
hook2: mov edi,Data_Buf_Addr
mov ecx,Inst_Data_Count
imul ecx,sizeof KM_InstData
add edi,ecx
push edi
add edi,sizeof KM_InstData
mov ecx,Data_Buf_Addr
add ecx,Data_Buf_Size
cmp edi,ecx
pop edi
jl hook1
mov ecx,Data_Buf_Size
add ecx,1000h
shr ecx,0ch
mov edx,Data_Buf_Handle
VMMcall _PageReAllocate <EDX, ECX, PAGEZEROINIT>
cmp eax,0
je hookerr
add Data_Buf_Size,1000h
mov Data_Buf_Handle,eax
mov Data_Buf_Addr,edx
jmp hook2
hook1: mov eax,calladr ;save caller's address in buffer
stosd
mov esi,Inst_Struc_Ptr
mov ecx,(sizeof InstDataStruc)/4
rep movsd ;save instance data struc
hookret:inc Inst_Data_Count ;next offset pair
pop eax
pop ecx
pop esi
pop edi
ret
hookerr:mov hook_err,-1
jmp hookret

EndProc Hooked_AddInstanceItem

```

End Listing

CONVERSE WITH YOUR PC and receive an intelligent reply!



Artificial Intelligence specialist Joseph Weintraub has won the Loebner Prize for Artificial Intelligence three years in a row. Weintraub is president of Thinking Software, Inc. In 1991, at the Boston Computer Museum, his PC Therapist became the first program in history to pass a limited Turing Test and win the Loebner Prize for Artificial Intelligence. PC Therapist convinced five out of the ten judges conversing with it that it was a human - not a computer program! Then in 1992

PC Professor, a program that talked about Women's Lib won the Loebner Prize again. Finally, in December of last year, Joseph Weintraub's PC POLITICIAN told Clinton a thing or two and won the prize for the third year in a row. PC POLITICIAN asks the provocative question "Are you a good solid conservative or a liberal piece of fruit?"

Now all this award-winning technology is available to run on your own PC. PC Therapist III is text only, and is \$59.95. In the animated PC Therapist IV a clever, bearded Therapist moves his eyes and mouth as he talks thru your PC Speaker....he will have you howling with laughter in no time for \$69 - for the smooth talking silicon buddy you've always wanted add \$18.95 for the SoundBlaster or Covox Version!

PC Politician is great for Presidents, Lawyers and Speech Writers! Yes, Bill, we have a free copy just for you....please drop us a note for your FREE PRESIDENTIAL COPY. All you other clods & freeloaders send \$119.95.

Includes FREE Expert Talking Demo Disk and 3 BIG Catalogs full of more unique software!

We always pay postage

Please specify disk size or we ship 3.5"

- Talking Demo Disk & Catalogs only send \$5 to

THINKING SOFTWARE, INC.

46-16 65TH PLACE - Dept XSX21

WOODSIDE, N.Y. 11377

Quality Software for the PC since 1986

AMEX, VISA, MC ORDERS
FAX (718) 898-3126
PHONE (718) 803-3638

CIRCLE NO. 897 ON READER SERVICE CARD

Listing Four (Listing continued, text begins on page 92.)

```
void CGAGraphDriver::Draw(CDC &dc, CWordMatrix &Grid)
{
    CPen *pPen = (CPen *) dc.SelectStockObject(BLACK_PEN);
    for (int row = 0; row < m_GridHeight; row++)
        for (int col = 0; col < m_GridWidth; col++) {
            if (Grid[row][col] != EMPTY_CELL) {
                int x1 = col * (CELL_WIDTH + CELL_SPACE) + CELL_SPACE;
                int x2 = x1 + CELL_WIDTH;
                int y1 = row * (CELL_HEIGHT + CELL_SPACE) + CELL_SPACE;
                int y2 = y1 + CELL_HEIGHT;
                dc.Ellipse(x1,y1,x2,y2);
                char buffer[4];
                sprintf(buffer,"%d",Grid[row][col]);
                dc.TextOut(x1+CELL_WIDTH/4,y1+CELL_HEIGHT/4,buffer,
                           strlen(buffer));
            }
        }
    //draw arcs
    for (int node1 = 0; node1 < m_NumGraphNodes; node1++)
        for (int node2 = 0; node2 < m_NumGraphNodes; node2++)
            if (m_pGraph->GetAt(node1,node2)) {
                int row1, col1;
                Grid.Find(node1, row1, col1);
                int row2, col2;
                Grid.Find(node2, row2, col2);
                int x1 = col1 * (CELL_WIDTH + CELL_SPACE) + CELL_SPACE;
                int x2 = col2 * (CELL_WIDTH + CELL_SPACE) + CELL_SPACE;
                int y1 = row1 * (CELL_HEIGHT + CELL_SPACE) + CELL_SPACE;
                int y2 = row2 * (CELL_HEIGHT + CELL_SPACE) + CELL_SPACE;
                if (x1 < x2)
                    x1 += CELL_WIDTH;
                else
                    if (x2 < x1)
                        x2 += CELL_WIDTH;
                else
                    if (x1 == x2) {
                        if (Abs(row1 - row2) > 1) { //route around!
                            y1 += CELL_WIDTH/2;
                            y2 += CELL_WIDTH/2;
                            int x3 = x1 - CELL_WIDTH/2;
                            dc.MoveTo(x1,y1);
                            dc.LineTo(x3,y1);
                            dc.LineTo(x3,y2);
                            dc.LineTo(x2,y2);
                            continue;
                        }
                        x1 += CELL_WIDTH/2;
                        x2 += CELL_WIDTH/2;
                    }
                if (y1 < y2)
                    y1 += CELL_HEIGHT;
                else
                    if (y2 < y1)
                        y2 += CELL_HEIGHT;
                else
                    if (y1 == y2) {
                        if (Abs(col1 - col2) > 1) { //route around!
                            if (x1 < x2) {
                                x1 -= CELL_WIDTH/2;
                                x2 += CELL_WIDTH/2;
                            }
                            else {
                                x1 += CELL_WIDTH/2;
                                x2 -= CELL_WIDTH/2;
                            }
                        }
                        int y3 = y1 - CELL_HEIGHT/2;
                        dc.MoveTo(x1,y1);
                        dc.LineTo(x1,y3);
                        dc.LineTo(x2,y3);
                        dc.LineTo(x2,y2);
                        continue;
                    }
                y1 += CELL_HEIGHT/2;
                y2 += CELL_HEIGHT/2;
                dc.MoveTo(x1,y1);
                dc.LineTo(x2,y2);
            }
        }
    dc.SelectObject(pPen);
}
//Calculate the length of the chromosome needed to encode
//a drawing of the graph in a grid
UINT CGAGraphDriver::CalcChromosomeLength() const
{
    return m_NumGraphNodes*(GetNumBitsToEncode(m_GridHeight) +
                             GetNumBitsToEncode(m_GridWidth));
}
UINT CGAGraphDriver::CalcRowAlleleLength() const
{
    return (UINT) GetNumBitsToEncode(m_GridWidth);
}
UINT CGAGraphDriver::CalcColAlleleLength() const
{
    return (UINT) GetNumBitsToEncode(m_GridHeight);
}
//Return TRUE if node1 is connected to node2
BOOL CGAGraphDriver::Connected(WORD node1, WORD node2) const
{
    return m_pGraph->GetAt(node1,node2);
}
//Returns the number of connection leaving node
int CGAGraphDriver::GetNumConnections(WORD node) const
{
    int count = 0;
    for (WORD i=0;i<m_NumGraphNodes;i++)
        if (i != node && m_pGraph->GetAt(node,i))
            count++;
}
```

```
return count;
}
//Returns the total number of connections in the graph
int CGAGraphDriver::GetConnectivity()
{
    int count = 0;
    for (WORD node1=0;node1<m_NumGraphNodes;node1++)
        for (WORD node2=0;node2<m_NumGraphNodes;node2++)
            if (node1 != node2 && m_pGraph->GetAt(node1,node2))
                count++;
    return count;
}
void CGAGraphDriver::Stop()
{
    m_Stop = TRUE;
}

Listing Five
//File: GRAPHGA.H
#ifndef __GRAPHGA_H__
#define __GRAPHGA_H__

//Headers needed for EOS programs
//You need EOS v1.1 to compile this code
#ifndef __BASISGA_H
#include "basisga.h"
#endif

class CGraphDrawerGA : public TBasicGA
{
public:
    CGraphDrawerGA(CGAGraphDriver &driver);
    void CreatePopulation(long size, PTIndividual prototype = NULL);
    void ExitReport();
private:
    BOOL Stop();
    void InterGeneration(ulong, PTIndividual, PTIndividual, PTIndividual,
                        PTIndividual);
    CGAGraphDriver & m_Driver;
};
#endif
```

Listing Six

```
//File: GRAPHGA.CPP
#include "stdafx.h"

//Headers needed for EOS programs
//You need EOS v1.1 to compile this code
#include "eos.h"
#include "geno.h"
#include "basicga.h"
#include "nptxgeno.h"
#include "genrepop.h"
#include "gaenviro.h"

//headers specific to graph GA
#include "gdriver.h"
#include "graphga.h"
#include "graphind.h"
#include "grphen.h"
#include "wmatrix.h"

CGraphDrawerGA::CGraphDrawerGA(CGAGraphDriver &driver)
: m_Driver(driver)
{
}
//Create the population of individuals
//We use 2 Point Crossover and Elitism
void CGraphDrawerGA::CreatePopulation(long size, PTIndividual prototype)
{
    //Create a genotype with 1 chromosome and 2 point crossover
    //The graph driver is queried to determine the chromosome length
    PTNPTCrossGenotype pGeno =
        new TNPTCrossGenotype(m_Driver.CalcChromosomeLength(),1,2);
    CGraphDrawingPheno * pPheno =
        new CGraphDrawingPheno(m_Driver,m_Driver.GetWidth(),
                                m_Driver.GetHeight());
    CGraphDrawingInd indiv(pGeno,pPheno);
    m_pPopulation = new TGenReplacePopulation(size,&indiv);
    m_pPopulation->SetElitism(2);
}
//When the GA is done set the best and worst individuals in the driver
void CGraphDrawerGA::ExitReport()
{
    m_Driver.m_pBest = m_pEnvironment->GlobalFittestIndivid;
    m_Driver.m_pWorst = m_pEnvironment->GlobalWorstIndivid;
}
//allow for windows processing!
void CGraphDrawerGA::InterGeneration(ulong, PTIndividual, PTIndividual,
                                     PTIndividual, PTIndividual)
{
    MSG msg;
    while (there are msgs for status window
           while (PeekMessage(&msg,AfxGetApp()->m_pMainWnd->
                               m_hWnd,0,0,PM_REMOVE)) {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
    SetCursor(LoadCursor(NULL, IDC_WAIT));
}
//GA calls this function to determine if it should stop
BOOL CGraphDrawerGA::Stop()
{
    return m_Driver.m_Stop;
}
```

End Listings



To kickstart PowerPC application development, Apple's APDA group has announced a number of Macintosh-based programmer tools for yet-to-come PowerPC-based Apple computers. The "Macintosh on RISC SDK" includes tools for creating new applications or porting existing Macintosh applications for future Apple PowerPC-based PCs. At the same time, Apple introduced the "Macintosh-with-PowerPC Starter Kit" and a comprehensive, self-paced training course entitled *Programmer's Introduction to RISC and PowerPC*. Additionally, Apple is offering Metroworks' native PowerPC development environment, CodeWarrior. Apple PowerPC-based computers are expected to become available in the first half of 1994.

The Macintosh on RISC SDK is an MPW-based cross-development environment that runs on a 680x0 Macintosh, generating native code for Macintosh-with-PowerPC-based systems. When these Macs become available, you can finish the port by testing and debugging your native Mac-with-PowerPC applications. The Macintosh on RISC SDK includes a C/C++ compiler that generates optimized code, PowerPC assembler, two-machine PowerPC debugger, universal system header files for both 680x0 and PowerPC processor-based platforms, MacApp 3.1 (Apple's object-oriented application framework), Apple Installer 4.0 (which is capable of installing either 680x0 or PowerPC environments from a common set of files), MPW Development System 3.3, a PowerPC linker, build tools and scripts, and sample applications for Mac-with-PowerPC.

The Macintosh-with-PowerPC Starter Kit includes detailed technical documentation about both the PowerPC microprocessor and System 7 for Macintosh with PowerPC. Among other information, this kit includes Motorola's *PowerPC 601 RISC Microprocessor User's Manual, Inside Macintosh: PowerPC System Software*.

CodeWarrior is a native development environment for the PowerPC-based

and 680x0-based Macintosh that lets you create applications for both platforms using the same source-code base. CodeWarrior comes in three versions: Gold, Silver, and Bronze. Gold, the most comprehensive, includes development releases of C/C++ for the 680x0 Mac and Mac-with-PowerPC, a development release of Pascal for the 680x0 Mac, and C/C++ cross-compilers. Silver supports native PowerPC development only, and will be released when Apple ships Mac-with-PowerPC systems. Bronze supports 680x0 development only.

The Macintosh on RISC SDK, available in prerelease with an automatic upgrade, sells through APDA for \$399.00, CodeWarrior Gold (also prerelease) for \$399.00, the PowerPC Starter Kit for \$39.95, and the Programmer's Introduction for \$150.00. Alternately, the tool sets are bundled, selling for \$849.00. Reader service no. 20.

APDA
Apple Computer
P.O. Box 319
Buffalo, NY 14207-0319
800-282-2732

A library of encryption tools implemented as linkable object modules and Windows DLLs has been released by AT&T. The library includes RSA, DES, El Gamal public-key, Secure Hash, MD5, and Diffie-Hellman encryption technology.

The code modules are packaged as SecretAgent (DES, El Gamal, and DSA digital signature), SecretAgent II (DES, RSA, and MD5), Surety (DSA), and SecureZmodem (DES using Zmodem protocol).

Prices for code packages containing DSA are \$750.00 for DOS/Windows, \$1000.00 for Macintosh, and \$1250.00 for UNIX. Packages that include RSA sell for \$300.00 for DOS/Windows, \$400.00 for Macintosh, and \$500.00 for UNIX.

The license allows programmers to load the code into two workstations for development purposes. Royalties are required for software distributed to end users. Reader service no. 21.

AT&T Secure Communications Systems
800-203-5563

Visual Xbase, a visual application-development tool for Xbase and C programmers, has been released by Rytech International. The tool includes a 3-D screen designer with an integrated, intelligent data dictionary, workbench, and a flexible, self-optimizing multidialect code generator.

Visual Xbase supports FoxPro 2.5 (DOS and Windows), Clipper 5.2, dBase (IV 2.0), and X2C, an add-on that converts Xbase code to C. In each case, the

tool generates code optimized to the selected language. It supports query by example, incremental table searches, filter by example, data-integrity searches, calculated fields, key checking, memory variables, cascaded deletions, and more.

Visual Xbase sells for \$495.95. There are no run-time license fees. Reader service no. 22.

Rytech International Inc.
2 Stamford Landing, #100
Stamford, CT 06902
203-357-7812

Mathematica toolkits for electrical engineering, signal processing, control engineering, statistics, finance, and similar disciplines are on the way. The first in the series is the Electrical Engineering Pack, which covers topics ranging from elementary to advanced and includes examples in circuit analysis, transmission lines, and antenna design. The EE Pack also covers Bode, Nyquist and root-locus plots, Smith Charts, and antenna-field patterns. New functions, which extend the original Mathematica product, are specifically aimed at the EE problem domain. Full source code for the examples is also included.

The Electrical Engineering Pack is available for the Macintosh, Windows, and X Window System and is priced at \$195.00. Reader service no. 23.

Wolfram Research
100 Trade Center Drive
Champaign, IL 61820
217-398-0747

Applied Cryptography: Protocols, Algorithms, and Source Code in C, by Bruce Schneier, has been published by John Wiley & Sons. In its coverage of cryptographic protocols, techniques, and algorithms, *Applied Cryptography* is perhaps the most complete book of its kind. For instance, Schneier (who is a frequent *DDJ* contributor) unravels virtually all block algorithms (including the NSA-backed Skipjack), public-key algorithms (from RSA to cellular automata), one-way hash functions, random-sequence generators, and special algorithms for protocols. In addition, the book provides over 100 pages of published source code for many of these algorithms. Likewise, Schneier's 35-page reference and bibliography section (listing over 900 sources) is of particular value for research.

The 618-page book sells for \$44.95 (ISBN 0-471-59756-2). Reader service no. 24.

John Wiley & Sons Inc.
605 Third Ave.
New York, NY 10158
212-850-6000

C CODE FOR THE PC

source code, of course

	GraphiC 7.0 (high-resolution, scientific plots in color & hardcopy, contour plots, device independence)	\$370
	X/DOS and Xt/DOS (Xlib with X client and Xt toolkit for DOS; port X code to DOS; Xt/DOS requires X/DOS and 32-bit compiler)	each \$300
	ZIP Image Processor & Victor Image Library Version 2.2 (brightness, contrast, merge images, TIFF/GIF/PCX/bin, HP ScanJet support)	\$290
	TE Editor Developer's Kit for Windows V3.5 (full screen editor, undo command, word processing; TER for application build-in; no royalties)	\$280
	C/C++ Libraries by Code Farms (persistent C structures, ER models, dynamic arrays, database functions, Jolt Award winner; specify C or C++)	\$250
	TurboTjX (Release 3.0; HP, PS, dot drivers; CM fonts; LaTjX; MetaFont)	\$250
	Rogue Wave tools.h++ or math.h++ Class Library (extensive docs)	each \$240
NEW!	Crusher! V2.00 (platform-independent data compression for network transfer; beats PK & LH on binary; directory trees; portable C)	\$215
	COMM-DRV (complete interrupt-driven serial communication libraries & device drivers; full source)	\$155
	TE Editor Developer's Kit Ver. 3.0 (full screen editor, undo command, multiple windows; with Word Processing \$230)	\$155
	Minix Operating System (Version 1.5; Unix-like operating system, includes manual; specify 5.25" or 3.5" diskettes)	\$150
	XASM (cross assemblers & utility programs; 65xx, 68xx, 80xx; Intel or Motorola hex format; macro preprocessor)	\$150
Updated!	Delorie GCC for MS-DOS (Version 2.2.2; includes C++, assembler, DOS extender, 387 emulation; complete source code and makefiles)	\$150
	Moby Crypto (PGP, DES, Secure Hash, UFC, MDs, Crack 4.1, Lucifer, IDEA, VCR+, large integer packs, tutorials, more; not for export)	\$150
	Lisp for DOS (Kyoto Common Lisp and CLISP; KCL includes Lisp-to-C translator for building mixed Lisp/C programs)	\$140
Updated!	Ibrow (Version 4.1; programmer's Windows-based editor; large files, help, undo/redo, drag-n-drop, function & type tags)	\$135
	SCM (portable Scheme in C, IEEE standard, includes JACAL symbolic math package; SCM-4D0/SLIB-1D5/JACAL-1A3)	\$100
	PC/IP (CMU/MIT TCP/IP for PCs; Crynwr drivers, NFS server, Bdale mailer, PCRoute/PCBridge, NDIS/ODI drivers, Beholder, more)	\$100
	DA (disassembler for Microsoft's New Executable (NE) binary files including Windows .exe, .drv, .dll, and .flt)	\$95
	Script Interpreter (a command script interpreter for DOS-based systems; C-like script language; lots of features)	\$90
	CELP 3.2c (Federal Standard 1016 Code Excited Linear Predictive voice sampling and encoding; voice over 4.8kbps; Unix code)	\$80
	CPPCOMM (C++ serial communications class library for DOS, Windows, OS/2, and NT; includes X/Y/Zmodem)	\$75
	ET Neural Net (back error propagation and Kohonen)	\$75
	FlexList (doubly-linked lists of arbitrary data with multiple access methods; specify C or C++)	\$65
	Kier DateLib (all kinds of date manipulation; translation, validation, formatting, & arithmetic)	\$60
Updated!	Coder's Prolog (Version 3.0; inference engine for use with C programs)	\$60
	PCCTS Version 1.10 (Purdue Compiler Construction Tool Set; like YACC and LEX together with lots of additional features)	\$60
	Container Lite V 1.82 (C++ & FLC wrapper emulators; portable, persistent containers of arbitrary data including pointers)	\$50
	BigFloat (arbitrary precision floating point arithmetic and functions; includes BCD conversion)	\$50
	EZCalc (ASCII algebraic expression evaluator, unlimited parenthesis nesting, symbols, 32 built-in functions, easily extended)	\$50
	Backup & Restore Utility by Blake McBride (multiple volumes, file compression & encryption)	\$50
	CLIPS Version 6.0 (rule-based expert system generator; Windows compatible; manuals on disk)	\$50
	SuperGrep (exceptionally fast, revolutionary text searching algorithm; also searches sub-directories)	\$50
	OBJASM (convert .obj files to .asm files; output is MASM compatible)	\$50
	Editor Pack (20 public domain editors; micromacs 3.12, Stevie, Elvis, Moke, mg2a, DTE, Jove, Origami, CE & GRIEF)	\$50
	Exceptions for C (Ada-like exception handling for C programs; exceptions for any block; exceptions can be reaised)	\$45
	DES Encryption & Decryption (2500 bits/second on 4.77 MHz PC for on-the-fly encryption at 2400 baud; not for export)	\$40
	Database Pack (9 databases - simple to complex: isam, bplus, AVL, SDB, ID, gdbm, Requiem, Ingres89, Postgres)	\$35
NEW!	COP (poor man's C++; C macro package which implements C++ in C)	\$35
	OCT (Object C Translator; essentially Brad Cox's Objective-C Version 4)	\$35
	RXC & EGREP Version 2.0 (Regular Expression Compiler and Pattern Matching; finite state machine from regular expression)	\$35
	Bison & BYACC (YACC workalike parser generators; documentation; includes C and C++ grammars)	\$35
	Spell Pack (6 spelling programs, a hyphenator, 2 utility packs and a 60K word list: Ispell, Microsp, Sp, Cspella, Spell, Dawg, Soundex)	\$30
	REGX Plus (Version 3.0, search and replace string manipulation routines based on compiled regular expressions)	\$30
	GNU Awk & Diff for PC (both programs in one package)	\$30
	Big Number Pack (7 arbitrary precision arithmetic packages in C, one in Fortran but free Fortran-to-C converter is included)	\$30
	Crunch Pack (30 file compression & expansion programs; now includes portable ZIP)	\$30
NEW!	OORT (C++ ray tracing code from the book by Nicholas Wilt)	\$30
	OEmacs (full GNU Emacs for DOS and Windows DOS box; C++ support, etags++, lots of .el files)	\$25
NEW!	CTask Version 2.22d (robust MS-DOS multitasking kernel; C functions run as light-weight processes; mailboxes, interrupts, pipes, etc.)	\$25
	PERL for MS-DOS (Version 4.019; C, sed, awk, and shell all rolled into one language; includes hardcopy docs)	\$25
Updated!	FLEX Version 2.4.3 (fast lexical analyzer generator; new, improved LEX)	\$25
	GNU RCS (FSF's version of the Revision Control System; like Unix's SCCS only better; keeps track of software development)	\$20
Data		
	Moby Thesaurus II (6,000 root words, 2.5M synonyms, "common sense", concept related searches)	\$500
	Moby Pronunciator II (175,000 words & phrases encoded with full IPA pronunciation & emphasis points)	\$265
	Moby Part-of-Speech II (230,000 words and phrases described by prioritized part(s)-of-speech)	\$170
	Moby Hyphenator II (185,000 words fully hyphenated/syllabified)	\$105
	Moby Words II (610,000 words & phrases with Scrabble(tm) word list, place names, baby names, acronyms, core list for spell checkers)	\$100
	CIA World Bank II Database (13MB of maps, 5.7M vectors; coastlines, rivers, political boundaries; 5.25" HD only)	\$35
	U. S. Cities (names & longitude/latitude of 32,000 U.S. cities and 6,000 state boundary points)	\$35
	The World Digitized (100,000 longitude/latitude of world country boundaries)	\$30
CD-ROMs		
NEW!	BSD/386 (POSIX-compatible O/S; complete development package, full networking, kernel debugger, X11R5, DOS box; complete source code)	\$900
	AI CD-ROM (expert systems, neural networks, genetic algorithms, fuzzy logic, linguistics/natural language)	\$105
Updated!	FontMaster II CD Library (soft fonts for HP and HP compatible laser printers, 36 different type faces; 5,200 bit mapped fonts; 300MB)	\$70
	Prime Time for Unix (Volume 3, No. 1, January, 1994; over 6GB of Unix C code)	\$60
NEW!	Walnut Creek Libris Britannia (over 600MB of the best of British boards; not all source included)	\$55
	Mailers Lookup (9-digit ZIP codes by street address or company name, distances between ZIP codes, phone locations; on-line tool)	\$50
	Linux/GNU/X by Yggdrasil Computing (1st production release; run from the CD; TCP/IP & NFS; drivers; MPEG; SCSI support; lots more)	\$45
	Walnut Creek C User's Group (Volumes 100 to 364)	\$40
	InfoMagic Unix (three public domain Unix systems: 386BSD (version 0.1), Linux (version 0.99.10), and NetBSD)	\$40
	InfoMagic Source Code (Berkeley Net/2, MACH, GNU, Interviews, X, Andrew, XFree, Demacs & Winemac, djgpp, Modula-3, etc.)	\$40
	Knowledge Media Multimedia (625MB & 13,000 files; 1,232 sounds, 179 books, 100 movies, 114 stacks, 606 programs, 214 mods)	\$35
	Project Gutenberg (literature, historical documents, reference books, census data, religious documents, math constants, etc.)	\$35
	Knowledge Media Languages & Operating Systems (640MB of compilers, libraries, and operating systems; source code & executables)	\$35
	Walnut Creek X11R5 and GNU (X11R5 with contributed and comp.sources.x, over 120 GNU programs, complete C source)	\$35
	Walnut Creek Usenet and Sintel Unix-C (600MB)	\$35
	Walnut Creek Giga Games (arcade, simulations, card games, education, trivia, cheat sheets; some source)	\$35
	Austin Code Works Internet Warrior #1 (PC Internet tools: Gopher, Wais, Eudora, ph, Nupop, Trumpet, TCP/IP, FAQs, drivers, docs)	\$35
NEW!	Walnut Creek FreeBSD (Berkeley 32-bit operating system for PCs; bootable)	\$35
NEW!	Walnut Creek Toolkit for Linux (Slackware distribution and complete Linux archive)	\$35
NEW!	Knowledge Media MegaMedia I and II (images, sounds, movies)	each \$30
	Walnut Creek CICA Windows Archive (August 1993)	\$25
	Walnut Creek Sintel 20 MSDOS Archive (C source code but lots of other stuff too)	\$25

The Austin Code Works

11100 Leafwood Lane

Austin, Texas 78750-3587 USA

much more ... ask for catalog

Voice: (512) 258-0785

FAX: (512) 258-1342

E-mail: info@acw.com

Free surface shipping for cash in advance

For delivery in Texas add 7%

MasterCard/VISA

(continued from page 148)

RenderWare, an interactive 3-D graphics API for Windows released by Criterion Software, supposedly increases Windows 3-D graphics performance, without the need for special 3-D graphics accelerators. Based on 3-D graphics software technology from Canon (Criterion's parent company), RenderWare reportedly enables mid-range workstation performance on a 486/50 PC.

RenderWare provides a device-independent 3-D graphics API, an object-based interface consisting of a small number of object types, and functions such as advanced shading and texturing. Typical applications for

RenderWare include multimedia, visual simulation, scientific visualization, CAD, virtual reality, presentation graphics, and entertainment/games.

In addition to Windows, the RenderWare API is available on other platforms such as Macintosh, UNIX (X11), and OS/2. The RenderWare SDK, which is priced from \$10,000.00, includes a development library, debugging library, documentation, examples, and demos. Reader service no. 25.

Criterion Software Ltd.
17-20 Frederick Sanger Road
Guildford, Surrey
United Kingdom, GU2 5YD
+44-483-574-325

Undocumented DOS, second edition, by Andrew Schulman, Ralf Brown, David Maxey, Raymond Michels, and Jim Kyle has been released by Addison-Wesley. The book, spearheaded by Schulman, who edits *DDJ's* "Undocumented Corner" column, has been updated to include coverage of MS-DOS 6, Novell DOS, Windows 3.1, the forthcoming "Chicago" operating system (DOS 7 and Windows 4), and more.

Like its predecessor, *Undocumented DOS*, second edition belongs on every PC programmer's bookshelf. The book, with disk, retails for \$44.95. (ISBN 0-201-63287). Reader service no. 26.

Addison-Wesley Publishing Co.
1 Jacob Way
Reading, MA 01867
617-944-3700

A set of tools that provide programmers with an API for fax-related applications has been developed by Sofnet. These tools, called the "FaxWorks API," facilitate the development of apps that integrate fax, OCR, scanning, voice, and image viewing. The API is proprietary, however, in that it was developed to support Sofnet fax-related software—FaxWorks Pro LAN, FaxWorks OS/2, FaxWorks ProServer, and so on.

The FaxWorks API forms a protocol layer in which other apps can exchange information. FaxWorks acts as a server when applications ask it to perform tasks such as faxing, scanning, or OCR. It acts as a client, however, when it asks applications for data such as a list of names and fax numbers from a phone book. The FaxWorks API and documentation are available free on CompuServe (GO SOFNET). Reader service no. 27.

Sofnet Inc.
1110 Northchase Parkway, Suite 150
Marietta, GA 30067
404-984-8088

As Ken North pointed out in his article, "Database Development and Visual Basic 3.0" (*DDJ*, March 1993), languages embedded in application programs are becoming more and more common. In the case of Microsoft, Visual Basic for Applications (formerly Object Basic) is a programming tool currently available only for Excel 5.0, but with more application support presumably on the way. WordPerfect has countered with WordPerfect 6.0 for Windows SDK and WordPerfect File Format SDK.

The WordPerfect 6.0 for Windows SDK features WordPerfect's new Writing Tools API, a macro language, and Shared Code 2.0 for Windows—a shared library of routines used by all WordPerfect for Windows software.

Application **SAVANT™** GENERATOR

Practical Solutions to the Programming Puzzle

Develop Windows Based Programs Without Extensive Training!

No compiling? Code free? No data libraries? Template free? No limiting functions? Unbelievable? Believe it! Never before has an Application Development System been this easy to use and this fast. Savant Application Generator software makes writing code an obsolete function of the software design process. You design the application's appearance and functionality on the screen with Windows™ and a mouse. Once the design of the application is laid out, anyone with word processing experience can create the application in a fraction of the time ordinarily involved using conventional programming techniques. Savant has the full functionality of a relational data base. With Savant you can create custom spreadsheets and custom reports which may be sorted in any order and generate an application map on demand. As you develop your application Savant allows you to test your algorithms and other formula functionality without having to compile. For further free information regarding

this exciting new development product please take a moment to circle the number below on the reader response card and mail it out today!

SAVANT
is a
revolutionary
Windows™
applications
development
tool for
computer
programmers.

Microsoft®
SOLUTION PROVIDER

© 1993 by Ariel Company. Savant is a trademark of Ariel Company. All rights reserved. Windows is a trademark of Microsoft Corporation.

PAXCELL
GROUP
Fax 702/831-1367

CIRCLE NO. 914 ON READER SERVICE CARD

The File Format SDK, which is available on a nondisclosure basis, contains documentation defining the WordPerfect 6.0 format and the WordPerfect Graphic File Format.

In related news, Softbridge has announced that SBL 3.0, an implementation of the Basic language for embedding into application software, now supports OLE 2.0 automation. This means that SBL, which has a syntax compatible with Visual Basic, can operate within and across applications.

The WordPerfect 6.0 for Windows SDK and WordPerfect File Format SDK sell for \$149.00 each. Reader service no. 28.

Softbridge provides various licensing arrangements for SBL 3.0. Reader service no. 29.

WordPerfect Corp.
1555 N. Technology Way
Orem, UT 84057-2399
800-451-5151

Softbridge Inc.
125 Cambridge Park Drive
Cambridge, MA 02140
617-576-2257

According to Al Stevens in this month's "Examining Room," good help can be hard to find when it comes to creating Windows help systems. Addressing this problem is Masterhelp, a new tool from Performance Software. MasterHelp takes text formatted for printing with Word for Windows and automatically creates a Windows help file, including hypertext jumps. The program also automatically creates Microsoft's Multimedia Viewer files for interactive tutorials and the like.

Among features created by MasterHelp are: a table of contents in a secondary window; a pop-up window which provides an overview of the entire document; a pop-up window that lets you know where you are in the document; and a pop-up window that shows hypertext-related topics. MasterHelp retails for \$495.00. Reader service no. 30.

Performance Software Inc.
575 Southlake Blvd.
Richmond, VA 23236
804-794-1012

Manageware for NetWare, a multiplatform tool for managing NetWare-based networks from Hitecsoft, is designed specifically for creating NetWare Loadable Modules (NLMs) and server-based applications. Manageware is based on a specification called the "Network Management Language" (NML) developed by Hitecsoft. Similar to a fourth-generation language, NML provides a more flexible means of accessing net-

work internals than traditional languages. NML is operating-system independent and has built-in network extensions (such as client/server, distributed processing, smart-object architecture, and others).

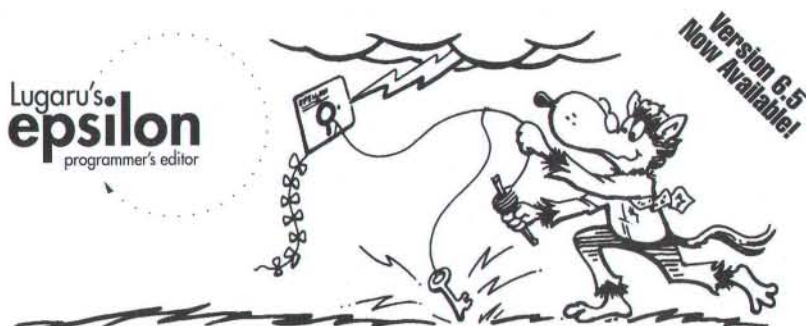
Manageware-based NLMs run under all supported platforms without source-code modification. This means that you can develop and test network-management programs under DOS, then run them as NLMs on the server.

Manageware Version 1.0 is an interpreter (and compiler for the developer's edition) that features a flexible preprocessor, virtual memory management, automatic variable declaration, exter-

nal function calls, and user-definable functions with local variable declaration and dynamic parameter passing. The tools provide full access to NetWare internals such as binderies, connections, directories, queues, IPX, SPX, and so on.

Manageware for NetWare sells for \$895.00. Reader service no. 31.
Hitecsoft Corp.
3370 N. Hayden Road, Suite 123-175
Scottsdale, AZ 85251-6632
602-970-1025

DDJ



Discover the Power!

Epsilon 6.5 costs \$250, with a 60-day money-back guarantee. You can order Epsilon for DOS, OS/2, SCO Unix, or Interactive Unix.

We are committed to continually improving Epsilon. When you purchase Epsilon, you can rest assured that we will continue to enhance it to take advantage of advances in hardware and software technology. We have been doing just that since 1984. Call us now at (412) 421-5911.

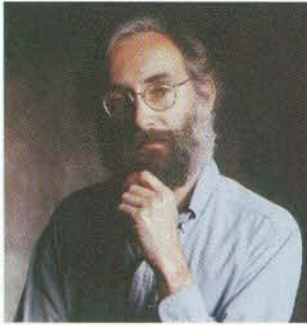
LUGARU
Lugaru Software, Ltd.
5843 Forbes Avenue
Pittsburgh, PA 15217
(412) 421-5911
(412) 421-6371 Fax

Epsilon Programmer's Editor combines a powerful and sophisticated command set with hardware savvy that takes full advantage of your PC's display modes, memory options, and other features, all so you can be more productive. With mouse support, a convenient concurrent process buffer, fast C tagging, enormous file capacity, and much more, Epsilon makes even your most arduous programming tasks manageable. So make tomorrow's coding easier—order Epsilon today.

Expanded Screen Modes Ever try to edit a 30-line function on a 25-line screen? With Epsilon, it's no problem to see the whole function. On any standard VGA board, Epsilon can display 25, 28, 35, 40, 43, or 50 lines on screen, and it's easy to add any other modes your board supports. And, unlike some other editors, Epsilon doesn't make you restart just to change modes—you can toggle between them with a simple keystroke.

Full XMS and EMS Support Epsilon fully utilizes either EMS or XMS memory, as well as upper memory blocks between 640kb and 1 meg. Epsilon will use that space for the text you edit, as well as for its commands, making it easy to edit even very large files. Why have all that memory and not use it?

Advanced EMACS-style command set Epsilon's comprehensive programming environment will make you more productive, because its well-integrated, extensive command structure does more in fewer keystrokes. For example, like some other editors, Epsilon can list files in a directory that match a pattern. But only Epsilon's commands work everywhere in the editor, so you can use Epsilon's sophisticated searching commands to locate the only file among thousands with two adjacent digits in its name. It's this kind of command integration that makes Epsilon the most powerful PC editor you can buy.



Pentium vs. PowerPC

It's Pentium vs. PowerPC. That's the simplistic view of what's going on in the area of personal-computer CPUs. DEC and MIPS Technology may see things differently, and this magazine is rarely simplistic about such matters, but this page is where we dumb down to the level of the rest of the computer press. Or even lower. And what's lower than a Lettermanesque Top Ten List? Generous to a fault, we give you two.

Top Ten Reasons why Pentium will Prevail

10. Anybody out there using the Dvorak keyboard? You do know that it's been shown to be superior in every way to the ubiquitous Qwerty keyboard, don't you?
9. On the PowerPC you'll have to run Windows and DOS apps under emulation. On Apple's own PowerPC machines, which it is calling Macintoshes, you'll have to run Macintosh apps under emulation. Emulation is slow. Emulation is an unnecessary layer of complexity. Emulation is evil.
8. The subliminal message. It probably wasn't a good marketing decision to call the technology behind the PowerPC "RISC."
7. Intel stock keeps going up.
6. In the short run, the ability to run DOS and Windows apps faster than a 486 machine is what will justify buying a new machine. In the short run, Intel wins.
5. In the long run, bet on the company with the deepest pockets. In the long run, Intel wins.
4. Everybody roots for the underdog. And puts their money on the favorite.
3. Compatibility. Compatibility. Compatibility.
2. Did I mention? Computer buyers value compatibility.
1. The Austin factor. Can we be absolutely sure that those Motorola guys won't withdraw the PowerPC from the market the first time *New York Times* columnist William Safire criticizes it? There's something in the water down there.

Top Ten Reasons why PowerPC will Prevail

10. It's got significantly faster floating-point performance than Pentium.
9. It's cheaper. By half.
8. Apple and IBM are solidly behind it.
7. Apple and IBM stocks are recovering.
6. If anyone is looking for a bridge from Intel to RISC, and a lot of people are, it's here. The first generation of PowerPC machines will run existing apps *under emulation* at speeds comparable to existing mid-range to high-end PCs. Native apps will be considerably faster. Early indications are that the emulations will be very solid.
5. Precedent. IBM's RS/6000 workstations haven't done too badly, and PowerPC is the migration of the RS/6000 processor technology to the personal-computer market.
4. The portable edge. The portables market is critical, and by releasing a Pentium chip that won't work in portables, Intel has given PowerPC a huge head start in portables.
3. Price. Price. Price.
2. Well, *do* computer buyers value compatibility? I mean, even if it costs them something? When have they ever had to pay for it? How much are they willing to pay for it?
1. The Clinton clincher. The future belongs to those willing to embrace change.

One reason that did not make the second list: It's Intel's turn to be the Evil Empire. No, IBM had the '80s and Microsoft gets the entire decade of the '90s. No honeymoon for Bill.

Michael Swaine

Michael Swaine
editor-at-large

C, C++ and BASIC programmers, now you get much more than xBase compatible DBMS power.

Thousands of programmers have already discovered how to get dBASE, FoxPro and Clipper compatibility from their favorite language and hardware platform. For example, one customer has C programs running on PC and Sun workstations sharing data with concurrently running FoxPro for Windows applications. You see, CodeBase technology is simply the best way to add multi-user xBase compatible DBMS power to C, C++, or BASIC.

You still get high speed & small size

CodeBase users really appreciate our small executable size. Unlike SQL engines which are a Meg or so in size, CodeBase 5.1 EXE's can be as small as 45K! You'll also like the speed—with our Intelligent Queries you get the execution speed of C plus stunning query performance from our smart use of available index information.

Introducing CodeControls

Now formatted data entry in Windows is as easy as point & click!

Introducing the new CodeControls, a unique set of data-aware custom controls. Now simply drop them into your Windows applications via your favorite visual interface builder.

NEW—You get formatted data entry

Experienced Windows programmers know formatted data entry is difficult

to program under Windows. But with our new **CodeControls**, you can simply 'Point & Click' to design data entry windows for date, numeric, and character information—formatted just the way you want it.

NEW—Data-aware controls

Our custom controls are *data-aware*, so now you can easily build a scrolling list box that's tied to a data file, or look up matching combo box entries—even as the user types.

Introducing CodeReporter 2.0

Now use the new Instant Report Wizard to create a variety of reports—instantly.

Introducing the new CodeReporter 2.0, our interactive xBase report writer. We re-designed it with developers in mind, but end-users will love it.

NEW—You get instant reports

Use our new Instant Report Wizard to quickly create a wide variety of reports—in an instant, then easily refine them as necessary, or let your end users build their own special reports!

NEW—You get drag & drop

We've added drag & drop capabilities to **CodeReporter**, so regardless of whether you want data, totals, or text, simply drop a report object onto your layout screen.

NEW—You get interactive queries, sorts, & relations

Use CodeReporter to visually design reports easier than ever. For example, you can quickly build query and sort expressions using our calculator-style expression builder. In addition, you can easily relate data files together using our graphical relation builder.



"There's never been a better time to buy into CodeBase technology. You get complete xBase compatibility, full DBMS power, plus an interactive report writer and a set of unique data-aware custom controls for Windows."

—Ken Sawyer, President, Sequiter Software

Buy One, Get Two FREE.

Now when you buy any one of our xBase library products: **CodeBasic**, **CodeBase++**, or **CodeBase** (for the language of your choice), you'll get both the new **CodeReporter 2.0** AND the new **CodeControls** absolutely **FREE**—for a limited time only.

To Order Now Call 403-437-2410

Unconditional 90-Day Money-Back Guarantee

SEQUIETER SOFTWARE INC. FAX 403-436-2999
UK Tel. +44-81-317-4321
France +33.20.24.20.14

P.O. Box 575 Newmarket NH 03857-0575



New Borland C++ 4.0 Visual is just the beginning

Why settle for ordinary visual when new Borland C++ gives you so much more? The highest quality fourth-generation tool set. A fully customizable and open desktop. The ability to target

16- and 32-bit Windows simultaneously. And, of course, it's "visual."

Only Borland C++ gives you a fully integrated professional editor featuring BRIEF® technology, powerful Turbo Debugger® GX, and integrated C and C++ VBX control support.

An environment to die for

Borland C++ 4.0 takes the bureaucracy out of development. With the visual Project Manager and its multi-target capabilities, even the most complex projects are handled automatically. The flexibility of AppExpert actually

generates much of your application for you. TargetExpert, ClassExpert, DialogExpert, and Resource Workshop™ streamline the tasks of setting up and customizing your applications.

A true C++ implementation that's years ahead

Languages evolve for a good reason—so programmers can realize ever-increasing productivity and safety of code. That's why Borland is the first to bring important new enhancements to the C++ language, like full support for templates, exception handling, and Run-time Type Information. And Borland C++ includes ObjectWindows™ Library (OWL) 2.0—the world's most popular framework.

Borland C++ is the world-standard C++ because it's the easiest to use, yet has power to spare for the most complex tasks. If you're serious about C++ programming, new Borland C++ 4.0 is the environment you've got to

have. After all, if your compiler is only visual, it's just a facade. Get new Borland C++ 4.0 for Windows and DOS now.

Borland C++ owners
Upgrade Now!

\$149⁹⁵
(Suggested list price \$499.95)

Other C++ owners
\$199^{95*}

90-day, money-back guarantee!
**See your dealer or call now,
1-800-336-6464, ext. 7160**
In Canada call, 1-800-461-3327.

CIRCLE NO. 95 ON READER SERVICE CARD

Borland
Power made easy™